Coalition Formation under Uncertainty

DISSERTATION

Daylond James Hooper

AFIT/DEE/ENG/10-05

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this dissertation are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

AFIT/DEE/ENG/10-05

Coalition Formation under Uncertainty

DISSERTATION

Presented to the Faculty

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

Daylond James Hooper, B.S.B.M.E., M.S.C.S.

March 2010

AFIT/DEE/ENG/10-05

Coalition Formation under Uncertainty

Daylond James Hooper, B.S.B.M.E., M.S.C.S.

Approved:

/signed/	15 Mar 2010
Gilbert L. Peterson, PhD Dissertation Advisor	date
/signed/	15 Mar 2010
LtCol Brett J. Borghetti, PhD Committee Member	date
/signed/	15 Mar 2010
Mark E. Oxley, PhD Committee Member	date

Accepted:

M.U. Thomas, PhD Dean, Graduate School of Engineering and Management	Date

Abstract

Many multiagent systems require allocation of agents to tasks in order to ensure successful task execution. Most systems that perform this allocation assume that the quantity of agents needed for a task is known beforehand. Coalition formation approaches relax this assumption, allowing multiple agents to be dynamically assigned. Unfortunately, many current approaches to coalition formation lack provisions for uncertainty. This prevents application of coalition formation techniques to complex domains, such as real-world robotic systems and agent domains where full state knowledge is not available. Those that do handle uncertainty have no ability to handle dynamic addition or removal of agents from the collective and they constrain the environment to limit the sources of uncertainty. A modeling approach and algorithm for coalition formation is presented that decreases the collective's dependence on knowing agent types. The agent modeling approach enforces stability, allows for arbitrary expansion of the collective, and serves as a basis for calculation of individual coalition payoffs. It explicitly captures uncertainty in agent type and allows uncertainty in coalition value and agent cost, and no agent in the collective is required to perfectly know another agents type. The modeling approach is incorporated into a two part algorithm to generate, evaluate, and join stable coalitions for task execution. A comparison with a prior approach designed to handle uncertainty in agent type shows that the protocol not only provides greater flexibility, but also handles uncertainty on a greater scale. Additional results show the application of the approach to real-world robotics and demonstrate the algorithm's scalability. This provides a framework well suited to decentralized task allocation in general collectives.

Acknowledgements

The sheer number of people that I should acknowledge cannot fit on this page. However, to name a few, I think it would be best to start with my advisor Gilbert L. Peterson, Ph.D. for his encouragement and confidence throughout. Starting with his willingness to be my advisor for the master's degree and continuing with his willingness to remain my advisor through my Ph.D. studies, his tolerance of my idiosyncrasies is commendable. Additional thanks is due for the many conversations, paper revisions, and day-to-day operations we endured. I must also thank LtCol Brett Borghetti, Ph.D. for his conversations in the early stages of my Ph.D. research. These conversations helped me to strengthen my approach and check all the little corners to ensure it was well developed thoroughout. Last on my list of academic people to thank is the ANT Center staff. The little bits of help here and there with the robots, general conversations, and occasional working lunches have helped me immensely.

My final thanks goes out to the most important contributors to my research: my wife and son. For her unending support, tolerance of randomness and absent mindedness, and eagerness to see me finished, my wife has earned my eternal gratitude. Due to his welcome distractions and insatiable curiosity, my son has helped keep me motivated. This degree is as much theirs as it is mine.

To everyone that I have not mentioned, you know who you are. Thank you for everything.

Daylond James Hooper

Table of Contents

Page

List of Figures

List of Tables

List of Abbreviations

Abbreviation		Page
CFDP	Coalition Formation Decision Process	3
INSeCT	Intelligent Navigation and Sensing for Cooperative Tasks	5
AFOSR	Air Force Office of Scientific Research	5
DAI	Distributed Artificial Intelligence	8
POMDP	Partially Observable Markov Decision Process	35
DEC-POMDP	Decentralized POMDP	35
I-POMDP	Interactive POMDP	35

Coalition Formation under Uncertainty

I. Introduction

Multiagent and multirobot systems provide redundancy, robustness, and the potential to perform more tasks than single agent and single robot systems [33]. An aspect of moving to multiagent systems is an increase in complexity. In particular, deciding which robot is going to fill which role for a given task, which is the task allocation problem. Task allocation mechanisms in cooperative systems tend to make one of two assumptions: tasks require only a single agent [17], or the number of agents required for a task are known *a priori* [21]. Those that do not make these assumptions can be classified in one of two ways: emergent or intentional. Emergent approaches allow assigning of multiple agents to a task, but the task generally requires a low degree of synchronization and the ability to assign agents at any point in the execution of the task. Intentional approaches involve dynamic coalition formation. Dynamic coalition formation reduces the constraints applied to a task and eliminates the assumptions about *a priori* knowledge of the agents required for a task through joining coalitions: subsets of agents formed to execute a task. This allows for greater flexibility in the collective's[1] execution and assignment of tasks. Additionally, it enables the collective to better respond to changes in its composition (i.e. the number and types of agents in the system). Many artificial intelligence and game theory researchers have performed work related to coalition formation [12,32]. Their research has created stable allocation schemes and mechanisms for forming coalitions.

One largely unaddressed area in coalition formation research is that of uncertainty [11]. This reduces the applicability of coalition formation to real-world systems, since deterministically defining the real world is prohibitively expensive in terms of processing and memory requirements. One way to bridge this representational gap

[1] A collective is defined as a collection of robots or agents, regardless of the coalition structure they form

is to use stochastic representations for the coalition formation related components. These components have different effects on the end result of the coalition formation process and the accuracy of the agents' estimates of the coalition's quality.

Other contributors to coalition formation are assumptions made either to employ a specific stability concept or to simplify the math. Among these are transferable utility and superadditivity. Transferable utility [30] is an assumption which improves a coalition's stability by allowing agents, players, or robots to transfer some currency between themselves. This transfer is lossless, and reduces the utility (payoff from the coalition) of one agent while increasing the utility of another. This functionality can also be considered currency transfer to keep agents in a coalition. This helps enforce stability, but is impractical for many real-world applications. Superadditivity [44] is an assumption made to simplify the math. It defines a larger coalition as better than a smaller coalition at all times. This serves to drive the coalition structure to only the grand coalition, then the math is simplified since merely distribution needs to be calculated.

One intent of the approach presented in this dissertation is to apply coalition formation to military systems. Due to the potentially long-lived nature of military systems, an approach to coalition formation that works for general, heterogeneous systems is useful. Unfortunately, the application of coalition formation to these systems is difficult since coalition uncertainty is not well captured. The components contributing to coalition uncertainty are the agent type (the physical construction and motivations of agents), agent cost (the resource expenditure incurred during task execution), coalition value (how well a coalition is expected to perform a task), and sensor noise. Approaches that allow for uncertainty in dynamic coalition formation are varied, and few capture the uncertainty explicitly. Vig and Adams [42] allow uncertainty in sensor noise, but neither capture it nor allow for uncertainty from any other source. Shehory and Kraus [35] allow uncertainty in agent type, but eliminate this uncertainty by requesting agent types at the beginning of a negotiation. Soh and Tsatsoulis [38] allow uncertainty from all sources, but assume that the agents in a

vicinity have full knowledge of each other. Chalkiadakis and Boutilier [11] allow for and capture uncertainty in agent type, but do not allow for uncertainty from any other source. This document presents an approach which improves upon the uncertainty captured by these coalition formation mechanisms by:

- Incorporating both game theory and distributed artificial intelligence work to provide a coalition formation algorithm that generates stable coalitions with neither transferable utility nor the superadditivy assumption.
- Explicitly capturing the uncertainty in agent type, agent cost, and coalition value through a unique agent model and evaluation algorithm.
- Providing an algorithm that solves for profitable and stable potential coalitions for a given task in a decentralized manner. The agents then form the individually most profitable stable coalition according to their local representation.

Traditional Coalition Formation Decision Processes (CFDPs) require knowledge of coalition payoffs, the agents potentially involved in the coalition (including the agent types), the coalition action or actions, agent costs, direct communication between agents, and that agents are members of only a single coalition. Unfortunately, these requirements constrain the application of the CFDPs to very well-defined environments. Additionally, there is no one common form of a CFDP that is applicable to an unstructured environment.

The work presented here also provides a formalization of CFDPs for coalitions with nontransferable utility designed to handle uncertainty. The formalization provides a more robust coalition formation mechanism for real-world robotic collectives, leading to reliable task-oriented multirobot systems. The CFDP employs a unique approach to generating and maintaining agent models and incorporating those models into a novel evaluation mechanism to solve for stable coalitions. It enables removal of many of the assumptions that have prevented more general applicability of coalition formation mechanisms. The agent model is composed of a vector of abilities, and these abilities provide insight into the suitability of an agent for a task. The

suitability (quality value) is used to determine an estimated agent and coalition cost and evaluate the cost relative to the payoff of the task. This provides insight into the overall estimated suitability of a coalition for a task. The approach also extends multi-robot task allocation and dynamic coalition formation in a manner that allows for dynamic changes in the environment, handles uncertainty, and permits dynamic change in the collective's composition. This allows for a heterogeneous composition and arbitrary addition or removal of robots or agents from a collective. The approach statistically provides results equivalent to other approaches in the face of uncertainty, and because the design framework is more flexible, it can solve harder problems. Additionally, the algorithm's execution time is similar in scalability to other algorithms for coalition formation.

1.1 Research Goal

The goal of this research is to formulate a CFDP in such a way that the representation includes provisions for uncertainty. The sources of uncertainty to be addressed include agent types, individual utility, and coalition value. The formation process should also include mechanisms and processing techniques that provide stability, scalability, and, at a minimum, optimality relative to agent beliefs. If these goals are met, the result should be applicable to real-world systems since the representation would allow for unstructured environments. The ensuing coalitions are stable based upon a selected allocation scheme. Additionally, the stability of the coalitions that are formed enforce a degree of robustness associated with any coalition action.

This research should also address the relationship between individual utility in a heterogeneous coalition and coalition value. Given the effects of individual quality on the value of a coalition, the payoff should be divided evenly according to the quality. The relationship is dependent upon the individual quality, and the application of a modified core for a rational payoff allocation based upon the relationship serves to stabilize a coalition further.

Demonstration of fulfillment of this research goal is shown by examinations of scalability, application to real-world robotic systems, and demonstration of the developed CFDP procedure's performance under uncertainty. The results in Chapters IV through V describe the experimental setup, testing, and results which indicate each of factors contributing to fulfillment of the research goal.

1.2 Sponsor

This research is part of the Intelligent Navigation and Sensing for Cooperative Tasks (INSeCT) project sponsored by the Air Force Office of Scientific Research (AFOSR). The associated autonomous navigation aspects for INSeCT include the need for the agents to dynamically form coalitions for cooperative tasks.

1.3 Assumptions

The collective of robots under consideration is lightly loaded, such that the processing time and resources needed to solve for stable coalitions are available on demand. This removes from consideration computation sharing (or processor time) as a currency for side payments.

The terms "transferable utility" and "side payments" are used interchangeably. Some authors make a distinction (for example, Aumann [4]), where they allow transferring of currency among players (side payments), yet any transfer does not affect the receiving player's utility (transferable utility). For the functionality under consideration here, the distinction between the two (i.e., side payments without transferable utility) is rhetorical.

Little distinction between agents and robots are made. If the term robot is used, it is generally assumed that it can apply to any embodied agent and that resources and hardware are treated similarly. Agents are assumed to have the same characteristics. If the term "disembodied agent" is used, it is treated as a purely software entity with

little or no analog to embodied agents or robots. Exceptions to this are noted where appropriate.

The agents or robots in a collective have a common communication language. The agents can communicate goals, state information, or coalition member-related data as necessary. However, certain data may not have a representation understandable by a given agent. This makes solicitation of agent types more difficult, since an agent may respond with an answer that the requester does not understand. In other words, an agent may represent another agent incorrectly and can only improve upon the representation through interaction with the other agent.

The agents or robots have a common task representation language. Under this assumption, each agent knows exactly what a task is composed of in terms of roles filled and any steps required to fulfill the task. Given this, the agent does not need to know everything about an environment. It is assumed that environmental knowledge (or lack thereof) does not affect the subtask sequence.

An abilities set can be reduced to a subset of abilities needed for task execution, and the dependencies between the abilities are known. In other words, the relationship of each ability to other abilities are known, and the task is processed such that the abilities required for each subtask can be generated.

1.4 Organization

This dissertation presents an improvement to previous approaches to coalition formation mechanisms with uncertainty. It provides an analysis of game theory, distributed artificial intelligence, and cooperative robotics work to illustrate the demand for more flexible and robust dynamic coalition formation methods. One method, including a formalized Coalition Formation Decision Process (CFDP) is presented in Chapter III. This is followed by the results of applying the approach to a simulated robot collective with variations in the agents' model accuracy and the collective size. The application of the approach to a real-world robotic collective is presented in

Chapter VI. Chapter V provides a comparison of the approach to another dynamic coalition formation approach designed to handle uncertainty. Appendices A and B provide a proof that the approach is immune to deadlock and an exploration of the agent model improvement approach (the modified Kalman filter), respectively.

II. Coalition Formation in Current Literature

Coalition formation is a popular area of research in cooperative game theory and artificial intelligence. The first direct address of cooperative game theory is in von Neumann and Morgenstern's book, Theory of Games and Economic Behavior [44]. They present and defend the concept of utility as a means to assign a value to a player[1] in a game. The primary focus of their work was on zero-sum games, games in which a gain for one player has an equal loss for its opponent. However, it is often advantageous to form a cooperative group to improve effectiveness by sharing workload and profit. This is the motivation for coalition formation research.

The research related to coalition formation can be broken into two areas: game theory and distributed artificial intelligence (DAI). Game theory coalition formation research focuses on coalition stability. This includes determination of rational allocations within a coalition and solving for stable coalitions. Very little effort is put into tractability and mechanisms, as these are the primary focus of the DAI aspects of coalition formation. An introduction to game theory coalition formation is covered in this document to provide a basis from which to present the DAI aspects that are used to create the final framework presented in Chapter III.

2.1 Game Theory

Some of the earliest work on stable allocation schemes was performed by Edgeworth [18], via a concept called the contract curve. This defines a line along which there exists an equilibrium between multiple parties. This was formalized into the modern understanding of the core by Gillies [23]. The core is the set of imputations (efficient and rational distributions) that cannot be improved upon by a coalition. The result is a rational[2] allocation of the profit for the coalition. A "coalitional form game" ν is defined as a function $\nu : 2^N \rightarrow \mathbb{R}$, where $N = 1, 2, ..., n$ is the set of all players (assumed finite). In mathematical terms, the core is where no coalition has a

[1] A player can be defined as any decision maker in a game, regardless of whether that player is human, computer, environment, etc.

[2] The term "rational" is used here since "fair" implies a subjective view of the allocation.

value greater than the sum of its members' payoffs:

$$C(\nu) = \left\{ x \in \mathbb{R}^N : \sum_{i \in N} x_i = \nu(N); \quad \sum_{i \in S} x_i \geq \nu(S), \forall\ S \subseteq N \right\}$$

where N is the set of all players, $\nu(S)$ is the value of coalition $S \subseteq N$, x_i is a payoff for player i, and x is the set of payoffs in the grand coalition that cover all players. This equation states that given any set of payoffs in the grand coalition, the sum of payoffs for each player is equal to the total payoff for the grand coalition and the payoff for each agent in the grand coalition is greater than or equal to what the player would receive in any other coalition. In other words, the core is the set of payoffs that the players cannot individually improve upon. The mechanisms underlying the core is best illustrated by an example. Consider 10 producers of item A and 11 producers of item B. The items are sold as a set AB, for a profit of \$10. A simple distribution of profit would be to split it evenly within each coalition of two producers. The core, however, considers the profit distribution based upon the ability to improve upon the current distribution. One of the item A producers can negotiate a better profit with the unpartnered producer of item B. A takes \$6, B takes \$4. This is an improvement over the previous coalition for both. However, this leaves the previous producer of type B with 0 profit since it is now unpartnered and cannot sell its item. It can improve its profit partnering with some other producer of A, which also improves its profit since it can increase its profit to 6 or higher by this coalition. This process would continue until all producers of A take all the profit and all producers of B take no profit. This cannot be improved upon by A nor B producers: the producers of A have improved their profit as far as possible, and the producers of B are doing no worse than they would if they were unpartnered. According to the core, the set of formed coalitions converges to this allocation. The core is very sensitive to oversupply, as indicated by this example. It is, however, rational and stable. The core does not guarantee optimality, if optimality is defined as the set of the most profitable coalitions. It does, however, guarantee an individually most profitable coalition. A

coalition formed for a specific task may be core-stable without being optimal, but the individual players in the core-stable coalition have achieved the best profit they can achieve. In a decentralized system with full environment knowledge, this is the full extent of the guarantees that an allocation method can make, regardless. Given this, some guarantees are available for k-optimality [8]. In essence, the core is a greedy, Nash-game style of allocation which enforces that the players are making the best decisions for themselves, not the entire system. The core is then well-suited to domains where the players may not act in a fully cooperative manner.

An allocation scheme not so sensitive to oversupply is the Shapley Value [44]. Consider a game v with a finite set of players $N = \{1, 2, 3, ..., n\}$. The total payoff of a coalition $S \subseteq N$ is $\nu(S)$. The value is an operator ϕ which assigns a vector of payoffs to each game ν, where ϕ_i represents i's payoff in the game. Under the assumption that $\nu(\emptyset) = 0$, player i's payoff in the game is:

$$\phi_i(\nu) = \frac{1}{n!} \sum_{\pi \in \Pi} [\nu(p_\pi^i \cup i) - \nu(p_\pi^i)]$$

with p_π^i as the set of players, $\pi \in N$ preceding player i in the order Π on S. The Shapley Value satisfies four axioms:

1. Efficiency: $\sum_{i \in N} \phi_i(\nu) = \nu(N)$. In other words, the resources available to the grand coalition are distributed among the players.
2. Symmetry: If any two players make the same marginal contribution to the coalition, both players have the same value.
3. Dummy: if player i is a dummy player, i.e., it makes no marginal contribution to the coalition, then $\phi_i(\nu) = 0$.
4. Additivity: the value is additive: $\phi(\nu+w) = \phi(\nu)+\phi(w)$ where the game $(\nu+w)$ is defined by $(\nu + w)(S) = \nu(S) + w(S)$ for all S.

An example of an application of the Shapley value is with two players. Define $\nu(1) = 1$, $\nu(2) = 3$, and $\nu(12) = 6$. Then the payoff for player 1 is

$$\frac{1}{2} \cdot (\nu(1) + \nu(12) - \nu(\emptyset) - \nu(2)) = \frac{1}{2} \cdot (1 + 6 - 0 - 3) = 2,$$

and for player 2, the payoff is

$$\frac{1}{2} \cdot (\nu(2) + \nu(12) - \nu(\emptyset) - \nu(1)) = \frac{1}{2} \cdot (3 + 6 - 0 - 1) = 4.$$

Note the satisfaction of the efficiency, symmetry, dummy, and additivity axioms.

There are two issues regarding the applicability of the Shapley Value to general coalition formation scenarios. The first is the superadditivity assumption, $\nu(S \cup T) \geq nu(S) + \nu(T)$ for all coalitions S and T such that $T \subseteq N$ and $S \cap T = \emptyset$. This implies that for any game, the best coalition is the grand coalition. Unfortunately, for real-world systems, as the profit is divided among participants, the highest profit for each player is often captured in a coalition smaller than the grand coalition. This is because coalition formation in real-world systems is not always fully cooperative. The second issue comes from both the dummy and symmetry axiom. Under uncertainty, it is difficult to determine the specific contributions made by a specific player. Declaring a player a dummy may limit the player's desire to participate in future coalitions, even if the contribution in those coalitions is significant. Additionally, any uncertainty associated with the player types limits the system's ability to reason about the marginal contributions of those players.

A third allocation scheme is the kernel [16]. The kernel is based upon bargaining power: the possible gains that a specific player can have by withdrawing from a specific coalition. The maximum surplus s (the maximum possible gain of a player by withdrawing from a coalition) for the game ν of player i over player j is

$$s_{ij}^{\nu}(x) = \max \left\{ \nu(S) - \sum_{k \in S} x_k : S \subseteq N \setminus \{j\}, i \in S \right\}$$

the kernel is the imputation set that satisfies

$$(s_{ij}^{\nu}(x) - s_{ji}^{\nu}(x))(x_j - \nu(j)) \leq 0$$

and

$$(s_{ji}^{\nu}(x) - s_{ij}^{\nu}(x))(x_i - \nu(i)) \leq 0$$

where x_i and x_j are payoffs for players i and j, respectively. As an example, consider a 3-person game with $\nu(\{1,2\}) = 90$, $\nu(\{1,3\}) = 80$, $\nu(\{2,3\}) = 70$ and $\nu(N) = 105$ [15]. Assume that players 1 and 3 have agreed *in principle* to a coalition with configuration

$$(x; S) = (45, 0, 35; \{1,3\}, \{2\})$$

where x is the payoff vector and S is the coalition structure. Here, the coalition {1 3} has agreed to a payoff of 45 to 1, 0 to 2, and 35 to 3. Coalitions {1} and {1 2} are coalitions that include 1 but exclude 3, so the value of these are determined in order to find s_{13}. $\nu(\{1\}) - x_1 = 0 - 45 = -45$. The value of coalition {1} is 0 and the payoff of player 1 in the current coalition structure is 45. The value for coalition {1 2} is $\nu(\{1\ 2\}) - (x_1 + x_2) = 90 - 45 - 0 = 45$. Thus, the maximum surplus of player 1 over player 3 is 45. Likewise, $s_{31} = 35$. Therefore, $(s_{13} - s_{31})(x_3 - \nu(1)) = (45 - 35)(35 - 0) = 350 > 0$, so this imputation is not in the kernel. The inequality $(s_{ji}^{\nu}(x) - s_{ij}^{\nu}(x))(x_i - \nu(i)) \leq 0$ need not be calculated. The primary advantage of the kernel is that it does not require superadditivity. Unfortunately, it also requires that all coalitions with a single member have a value of 0 for most applications. This approach allows for uncertainty in the most straightforward manner, since it reduces the uncertainty to coalition values only. Additionally, the time for computing an optimal coalition under these constraints is $O(N^N)$ [34], once the kernel is found, which itself takes exponential time [7]. The $O(N^N)$ time is based upon the total number of coalition structures:

$$\sum_{i=1}^{N} Z(N, i)$$

where $Z(N,1)$ is the number of coalition structures with i coalitions and $Z(a,i) = iZ(a-1,i) + Z(a-1,i-1)$ [34]. This leads to the n^n runtime for determining the coalitions. For each of these, the kernel is determined, making for significant calculations. This is impractical for most applications of coalition formation due to these time constraints.

2.2 Uncertainty in CFDPs

Coalition formation decision processes (CFDPs) are traditionally assembled with assumed knowledge of coalition value, agent types in the coalition, the coalition action, and agent costs. The usefulness of traditional CFDPs is therefore constrained to well-defined environments. In real-world robotic systems, the environment is often poorly defined, preventing application of CFDPs to the system. One way to bridge this representational gap is to use stochastic representations for the CFDP-related components. The remainder of this section is broken up according to the type of uncertainty: agent type, agent cost, or coalition value. Section 2.2.1 presents previous work related to uncertainty of agent types. Section 2.2.2 discusses agent cost uncertainty. A discussion of work related to uncertainty in coalitional value is provided, as well as the relationship between each of the three sources of uncertainty. The relationship is cyclic, as one source of uncertainty can lead to uncertainty in other sources. Furthermore, feedback on the cycle can improve uncertainty over repeated formation attempts. These relationships are shown in Figure 2.2.

2.2.1 Agent Type Uncertainty. Agent type is a key factor in determining the quality of a potential coalition. Agent type describes an agent's abilities and motivations, allowing for reasoning about the contributions of the agent to a potential coalition. Unknown agent types inhibit the quality of the coalition value and agent cost estimates, as the three are mutually dependent. Stability[3] of a coalition can still

[3]Stability is defined as whether a coalition will remain in existence. A coalition that may dissolve is unstable, one that may not is stable.

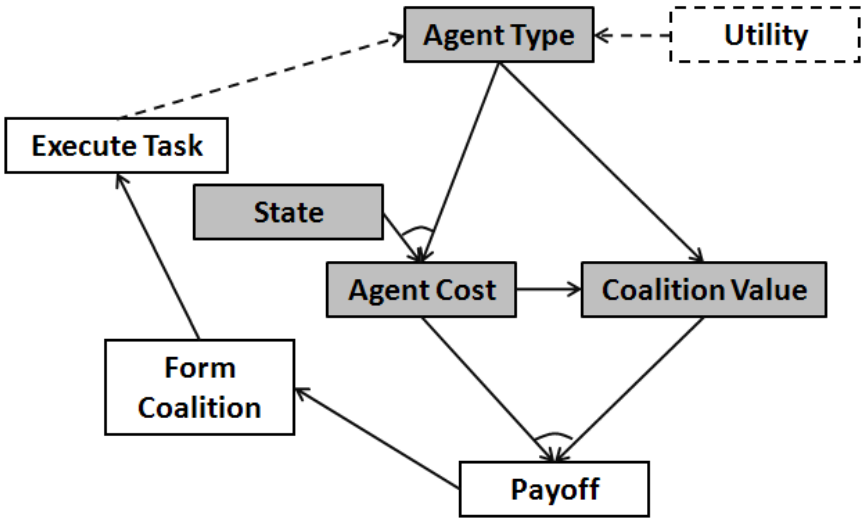

Figure 2.1: AND/OR graph showing dependencies in coalition formation processes. The darker boxes are sources of uncertainty. The arrows indicate the direction of the dependency. Dashed lines indicate temporal values, fed in at the beginning or end of the formation cycle. The quality value input and execution result feedback are unique to the algorithm presented in this paper.

be guaranteed based upon agent estimates: if the agents form a coalition and expect that it is the best according to some stability criterion, the coalition is stable. This is the full extent of the available guarantees under uncertainty, and one that is discussed later in this section. The work related to uncertainty in agent types presented in this section assumes that either the agent type information can be obtained or learned, which drives the coalition formation mechanisms to an approach where estimates define the agent types and, thereby, the underlying stability.

Shehory and Krauss [35] provide a mechanism that does not explicitly require *a priori* information about the agent types, but solicits this information from the other agents during a CFDP. Their work assumes transferable utility (a concept discussed in depth in Section 2.2.2), which is infeasible for many real-world robotic systems. The environment for which their mechanism is designed does not have the property of superadditivity, which coincides with the environmental view of a real-world robotic collective. Based upon the non-superadditivity assumption, the grand coalition is not

necessarily the most beneficial. The property of superadditivity is central to the use of the Shapley Value, so their framework is only applicable to the core or the kernel. Their assumptions allow a coalition to focus on a common task, since each coalition can work on only one task at a time. An agent may execute multiple simultaneous tasks by joining multiple coalitions. Shehory and Krauss apply an algorithm to solve for all possible coalitions and perform a greedy selection of the preferred coalitions to determine the best coalition structuring. The resulting coalition structure is not guaranteed optimal due to the greedy selection strategy, though guarantees can be made regarding k-optimality [8]. Once the agent type information is solicited, there is no further uncertainty in agent type. Given this information and the knowledge of agent costs and coalition value, the uncertainty is not incorporated into the algorithm directly. A problem with this representation is if an agent is of a type that another agent cannot accurately represent. Dynamic changes in the composition and increased heterogeneity in the collective require reprogramming of all of the agents in the collective.

Chalkiadakis and Boutilier [11] fully incorporate uncertainty in agent type, assuming that neither the solicitation of agent types nor prior knowledge exist. Instead, the agents learn the types of the other agents by repeated coalition formation. The uncertainty in agent types is directly addressed by their work, though the set of possible agent types is known. The most significant contribution of their work is the Bayesian core stability concept. The agents are required to determine coalitions based upon their individual beliefs. This does not guarantee that a coalition is formed. In fact, the Bayesian core may be empty even when the core (if all types are known) is nonempty, since the beliefs might not reflect reality. If the beliefs are sufficiently inaccurate relative to the true values, a poor coalition may form. The Bayesian core reflects this, since it is built upon beliefs. Let $N = \{1, 2, ..., n\}$ be the set of agents in the collective. Agent i's beliefs about agent j's expected payoff if j is a member of any coalition $S \subseteq N$ with coalition demand vector (the relative payoff demands for

the coalition) $\vec{d_S}$ taking coalition action α is

$$p_{j,S}^{-i}(\alpha, \vec{d_S}) = \frac{d_j Q_i(S, \alpha)}{\sum_{k \in S} d_k}$$

where $Q_i(S, \alpha)$ is the expectation of the value of action α taken by coalition S. This equation relates the demand d_j, and expectation of action values for coalition S taking action α, $Q_i(S, \alpha)$ (Q-value), to the total demand of the coalition members, $\sum_{k \in S} d_k$. Thus, with a high Q-value, low total demand, and high demand for j, i believes j would expect higher payoff from coalition S.

By this definition, a coalition structure CS with demand vector $\vec{d}$ is in the Bayesian core (BC) if and only if all coalitions in the structure have an action such that there is no alternative coalition that has an action with expected higher payoff. In mathematical terms,

$$\langle CS, \vec{d} \rangle \in BC \text{ iff } \forall S \in CS \ \exists \alpha \in A_S \text{ s.t. } \forall C \subseteq N$$

$$\nexists \beta \in A_C \text{ and } \vec{d_C} \text{ s.t. } p_{i,C}^{-i}(\beta, \vec{d_C}) > p_{i,S}^{-i}(\alpha, \vec{d}), \forall i \in C$$

This definition is separated into Bayesian cores of differing strength values in [14]. There, the weak Bayesian core, the strict Bayesian core, and the strong Bayesian core are defined. The weak Bayesian core is when there is no coalition in a coalition structure where all members believe that they can be personally better off. If there is a member who proposes an alternate coalition allocation that requires all members to change some aspect of their allocation, even one member responding that they can't be any better off than they currently are (in expectation) restricts the coalition to the current structure. This is still in the Bayesian core and stable, but readily becomes unstable when, for example, there is only one agent that disagrees with the alternate allocation, but then changes its mind.

The strict Bayesian core, a subset of the weak Bayesian core, is where there is no agent in a coalition who believes that an agreement that makes it better off without

hurting the other members of the coalition exists. In other words, if there is one agent who believes it can do better but has to do so at the expense of the other agents, the alternative agreement does not occur. This perspective is from an omniscient point of view: the determination of the hurt to j is true harm to j, not an estimate from i.

The strong Bayesian core takes the strict Bayesian core a bit further, assessing beliefs about the other agents. The strong Bayesian core is where there is no agent in a coalition who believes that an agreement that makes it better off without expecting to hurt the other members of the coalition exists. The subtle difference between this and the strict Bayesian core is that the determination of the hurt to the other members of the coalition from agent i's perspective is based upon i's estimation of the payoff for player j, instead of player j's estimation of its own payoff. In other words, if there is one agent who believes it can do better but has to do so with an expectation that it would be at the expense of the other agents, the alternative agreement will not occur.

Chalkiadakis and Boutilier expand upon a means to address this uncertainty [13] using repeated coalition formation to apply reinforcement learning in a more concrete sense than in [11] to learn the agent types. The drawbacks of the framework they present are threefold:

1. The true agent types might not match one of the predefined agent types.
2. The agent types, even if known, may not be enough to determine the cost associated with the agent and, in turn, the coalition value.
3. The application to real-world robotics is limited due to the underlying assumption of transferable utility with the agents.

The work they present does directly address uncertainty with agent type, however, and is useful as a basis for incorporation of other sources of uncertainty. Agent type is the first source of uncertainty encountered, and since it is used in part to calculate the agent costs and coalition values, can contribute the most to the formation of poor coalitions if not well addressed. It is also the most straightforward to handle,

since agent type is independent of the environment. Agent cost, however, is not as straightforward. It is discussed in the next subsection.

2.2.2 Agent Cost Uncertainty. The costs of each agent directly determine the quality of the agent in a coalition. Individual quality may have some uncertainty associated with it, and the approaches to addressing it are limited in their applicability. In many Game Theory examples, the existence of Transferable Utility (TU) is assumed. TU exists in scenarios where there is some common currency that is suitable to make side payments within a coalition [25]. This is impractical for many situations, since the currency may be nontransferable. A real-world mobile robotic collective would not have TU, since the common currency, be it energy, hardware, or processing time[4], is not transferable. This is not to say that social rules could not be applied. If a means is in place to enforce social rules on transferable utility, these transfers can be made later. However, most practical and robust systems avoid application of rules in this way. These and other similar situations have Nontransferable Utility (NTU). A coalition with NTU is often less stable than one with TU. In NTU coalitions, the appropriate characterization of each individual agent's quality is of crucial importance. For example, an agent with very low quality in a coalition that expects high payoff is of little worth to the coalition and may end up costing the coalition, rather than contributing to it. This is handled readily in TU scenarios, but increases complexity in NTU scenarios. The individual quality is critical to capture in order to allow for reasoning on the coalition under consideration.

Kargin [29] presents uncertainty as it pertains to agent costs in a Shapley Value stability concept application. The Shapley value has a superadditivity assumption, which implies that the most valuable coalition is one with all agents involved (or one where agents can at least guarantee their value). The determination of the marginal contribution of each agent serves to define the distribution of coalition rewards to the

[4]We assume the robots are lightly loaded, i.e. the processing time and resources needed to solve for stable coalitions are available to each individual on demand.

coalition. The uncertainty Kargin presents is that of an agent in a random coalition. Let $f_i : X \rightarrow 2^{(N-\{i\})}$ be a random variable with its values in the set of subsets of $N - \{i\}$. Define the marginal contribution of player i as $d_i G$. Then the Shapley value V_i of game G for player i is the expectation $\mathbf{E}(\cdot)$ of the form

$$V_i(G) = \mathbf{E}[d_i G \circ f_i]$$

which is the expectation of player i's marginal contribution when the coalition is chosen randomly. The Shapley uncertainty R_i for player i of this game is the variance $\mathbf{Var}(\cdot)$ of the form

$$R_i(G) = \mathbf{Var}[d_i G \circ f_i]$$

which is the variance of the marginal contribution of player i to the game. This uncertainty is applicable to games where the marginal contribution to the grand coalition game is known. In other words, the agent cost (higher cost implies lower utility and vice versa) for the grand coalition is known, and the agent cost for a given coalition is known (or is readily determined). The agent cost for a random coalition is characterized by the uncertainty of the Shapley value, meaning that once a coalition is realized, the uncertainty is irrelevant. The mechanism provided here by Kargin serves to provide a means to characterize the value of an agent to a random coalition when the agent's value in the grand coalition is known. This provides insight into how to characterize the contribution when it's not known, but this work by Kargin is incomplete in terms of fully capturing the uncertainty in agent utility. Kargin's work serves to indicate uncertainty in a randomly chosen coalition when the agent qualities are fully known.

Uncertainty in agent cost is largely unaddressed, as the assumption of transferable utility serves to improve stability by "forcing" a cost onto each agent in a coalition. The transferable utility assumption also makes any formed coalitions more stable. Unfortunately, the application of transferable utility to real-world robotics in terms of coalition formation is limited, and, from the arguments presented above, not

applicable to the domain. In fact, the assumption of TU has been called "exceedingly restrictive–for many purposes it renders n-person theory next to useless" [30]. Directly addressing this uncertainty provides some insight into the uncertainty in both agent type and the overall value of the coalition. The next subsection discusses uncertainty in the coalition value and presents an approach to addressing it.

2.2.3 Coalition Value Uncertainty. In addition to individual quality, a source of uncertainty is the value of a coalition. The payoff associated with a coalition may be known, but given a number of potential coalitions that can perform the associated task, the value of each coalition determines its suitability for the task. This is best illustrated by an example: movers are paid a set amount to relocate a piano. For the sake of simplicity, assume payment is divided evenly among the participants in the move (i.e., each mover has the same utility or cost). The movers must maximize individual payoff along with minimizing expense. Assume that it takes a minimum of two movers to relocate the piano. Assuming equal capability of each mover, the optimal coalition size is two: it maximizes profit while avoiding unnecessary expense in attempts with a singleton coalition and overallocation with the grand coalition. In this example, the value of any possible coalition is known. If the example is modified such that it is not known how many movers it takes to relocate the piano, the coalition value becomes uncertain. The coalition payoff is known, but the costs and payoff for each mover is unknown, thus establishing an unknown coalition value. The unknown coalition value creates an additional source of uncertainty, which must be reasoned about. Note that, if the movers are not equal in capability, individual utility dictates the coalition value and the division of the payoffs. This relationship is important, as it may cause instability. This instability is normally addressed by TU in other coalition formation work.

Blankenburg *et al.* [7] present a coalition formation technique based upon the kernel stability concept that incorporates uncertainty in the coalition value. The kernel is based upon the coalition values alone. If the values of all potential coalitions

are known, the kernel provides a stable and rational allocation. Even if the values of the potential coalitions are uncertain, they are characterized and captured by fuzzy sets in Blankenburg's work. The fuzzy sets are applied as membership functions to determine the maximum fuzzy excesses of a given agent over another agent. This is used, in conjunction with a fuzzy ranking operator value and fuzzy similarity relation, to determine the bargaining power one agent has over another. The kernel stability concept is applied to the bargaining power determination to form stable coalitions. Blankenburg's work is useful in that the assumption of transferable utility is not implied. The fuzzy set approach implies that some *a priori* knowledge is available that indicates a membership in a small number of possible maximum surpluses. Additionally, the kernel is solely concerned with coalition value. The contributing factors, agent cost and agent type, affect the coalition value and is not captured by Blankenburg's work. The relationship between these factors is discussed in the next subsection.

2.2.4 Relationship Between Uncertainty Sources. The kernel, core, and Shapley value each have different required knowledge. For example, the kernel is only concerned with coalition value, and the core and Shapley value are based upon agent quality. If full knowledge of the information required by each stability concept is present, the uncertainty in the other sources need not be captured. Conversely, full knowledge of the information not required by the stability concepts provides only minimal insight into the information they require. Unfortunately, the three sources of uncertainty are related. If the agent type is unknown, the value of the agent to the coalition is unknown, in turn making the coalition value unknown. Thus, each stability concept is dependent upon all three uncertainty sources, though some of the dependence is indirect. An aspect of the three sources of uncertainty is that knowledge of one does not necessarily imply knowledge of the others. Chalkiadikis and Boutilier [11] use agent type uncertainty to determine the core. Unfortunately, the agent type and agent utility are not as closely related in real-world applications as their

approach implies. Knowledge of the agent type provides little insight into the agent cost and coalition value since environmental factors may be unknown. Conversely, knowledge of agent costs for all agents in the coalition implies knowledge of the coalition value. Failing to know the cost of even one agent in the coalition prevents knowledge of the coalition value, in turn making the computation more difficult since that agent's influence is unknown [2].

As an alternative to handling uncertainty, it can be deflected, choosing instead to indirectly address the unknowns. Soh and Tsatsoulis [38] provide a representation that allows for uncertainty with some agents, but requires adequate *a priori* knowledge of the agents in a neighborhood. This reduces the complexity of the calculation to the well-understood neighborhood. There is not, however, uncertainty associated with the agents in the neighborhood, thus, on this neighborhood, the CFDP is free from uncertainty. One benefit of their work is that it allows for membership of one agent in multiple coalitions simultaneously. Additionally, their work incorporates the potential for communication failure during CFDPs. The coalition formation mechanism they present relies heavily upon negotiation, maintaining and managing multiple negotiation threads. There are three key aspects to the negotiation: choosing whether to negotiate, available negotiation threads, and availablility of communication channels. A prevention of any of these three aspects causes a failure of the coalition formation. This mechanism in practice was only successful 20% of the time. The advantage of their work is that it provides reasoning and decision making based upon beliefs: the knowledge of the other agents in a neighborhood creates beliefs about those outside the neighborhood. The intended application for their work is a stationary multisensor network, so those sensors that are in a neighborhood are most likely to see the same data and most likely to benefit from forming a coalition. Those outside the neighborhood are less likely to see the same data, thus forming a belief about the other agents. The full knowledge of the other agents in a neighborhood, however, limits the applicability of this mechanism to a more uncertain collective.

Table 2.1: Comparison of the uncertainty sources handled (agent type, agent cost, coalition cost), transferable utility vs. nontransferable utility, and valid allocation schemes for the works reviewed in this section.

	Uncertainty				
Author	**Agent Type**	**Agent Cost**	**Coalition Cost**	**TU/NTU**	**Schemes**
Shehory [35]	solicit	no	no	TU	core, kernel
Chalkiadakis [14]	yes	no	by proxy	TU	core
Kargin [29]	no	yes	no	TU	Shapley Value
Blankenburg [7]	no	no	yes	TU/NTU	kernel
Soh [38]	no	no	no	TU	kernel
Vig [42]	no	no	yes	TU	kernel

Vig and Adams [42] make use of the algorithm from Shehory and Krauss [35] in a multi-robot setting to form coalitions. The uncertainty in their approach is limited, however, and not directly addressed. Sensor noise and communication failure is common to all real-world robotic systems and creates a degree of uncertainty associated with the robot's internal state estimate. However, the types and capabilities are assumed to be known *a priori* in their work. This may be a valid assumption for static robotic collectives where the coalitions are a subset of the collective, but collectives that allow for the addition of new (previously unencountered) robot types cannot use this assumption. They provide an optimization of the greedy algorithm which determines the lopsidedness of a coalition. Their approach considers the contributions of each potential member of the coalition and attempts to equalize them. They posit that one coalition with large contribution from a single member is more likely to fail, since the single, highly capable member enables a single point of failure. This approach also does not consider the cost of forming a coalition, nor does it attempt to minimize the number of members in a coalition. This makes a more stable coalition structure, but reduces the individual payoff in the formed coalitions.

Table 2.1 shows a comparison of the approaches from each of the items reviewed in this section. With the exception of Blankenburg *et al.*, which makes no assumption, the commonality of these approaches is that they all assume transferable utility. There are some approaches that are designed to handle coalitions that cannot make

sidepayments, i.e., they have nontransferable utility. This is discussed in the next section.

2.3 Coalition Formation with Nontransferable Utility

The previous section discussed general coalition formation mechanisms and ways of incorporating uncertainty into them. Unfortunately, the mechanisms that were presented were only defined for situations that require the existence of transferable utility. The application of coalition formation to real-world collectives is limited because NTU is very common in these collectives. This section provides a discussion of mechanisms for coalition formation in collectives that have NTU.

The relationship between TU and side payments is subtle, but distinctive for certain applications. Non-cooperative games contain cooperative games as a subset, and cooperative games without side payments include cooperative games with side payments as a subset (in other words, any solution method used to solve cooperative games without side payments is applicable to cooperative games with side payments). Transferable utility is a special case of cooperative games with side payments [5]. Whenever the games are addressed directly, the authors rarely make it a point to draw a distinction between cooperative games with neither side payments nor TU and cooperative games with side payments but without TU. Those that do directly address the distinction normally apply their work to the former, since it is more general. Due to the applicability of the results to games with side payments, this is the assumption applied here. In fact, transferable utility and side payments are assumed here to be the same thing, since normally the assumption of the presence of one implies the presence of the other.

Tohmé and Sandholm [40] present coalition formation processes in a setting with self-interested agents. These agents are boundedly rational. In order to establish an appropriate framework for the game at hand, the α-assumption is used. This assumes that all nonmembers of a coalition make decisions that are the worst for the coalition under consideration. This pessimistic view enables a "worst case" coalition to be

formed. This worst case coalition structure is called the α-core, which is the stable coalition structure formed under the α-assumption. Given that the agents are self-interested and the coalitions are making the worst possible decisions for each other, the game is naturally reformed into a Nash-style game. In other words, the Nash game defined here answers the question, "given this worst case strategy of the other coalition, what is my coalition's best strategy?". Additionally, since the core is the stability concept applied, the agents have a Nash game between individuals. Tohmé and Sandholm assume superadditivity, so, though their framework does allow for the removal of the TU assumption, the superadditivity assumption removes much of the processing associated with determining the coalition structure. The optimal coalition structure is the grand coalition according to this assumption, so the ideas of multiple coalitions and agent membership in multiple coalitions is not addressed. The stability of the coalition is also not addressed due to the superadditivity assumption. The framework they present allows for different actions within a coalition, so the Nash game is defined traditionally: an agent determines its optimal actions after evaluating the possible actions of the other agents. After the action is performed, it revises its beliefs about the actions of the other agents in order to better estimate the future actions of others. Their approach indicates the Pareto-optimality of their utility profile: to improve individual utility, the utility of another agent must be reduced. This is useful to note, as it still has the underlying properties of a common currency. For real-world robotic systems, these utilities are static (though unknown or uncertain). Tohmé and Sandholm's solution draws heavily on the α-core and the α-assumption. These were presented by Aumann and drawn upon in some of Aumann's own work [4].

Aumann provides a survey of cooperative games without side payments in [5]. He clarifies the definition of the characteristic function, which is a simple representation of a game. The characteristic function gives the set of payoff vectors for each coalition that the coalition can assure its players. One can define a game using the characteristic function as a pair (v, H) where $v(S)$ is the set of payoffs that a coali-

tion S can assure itself (the characteristic function) and H is the set of outcomes that can occur[5]. He describes three conditions for this definition. The first condition is that the characteristic function is closed, convex, and non-empty. This condition holds intuitively: players can mix their strategies (convexity), the vectors in the set are attainable (closedness), and the coalition S is only formed if there is some payoff associated with it (non-emptiness). The second condition he applies is that if a coalition can assure itself a given payoff vector, it can also assure itself anything less. The third condition he applies is superadditivity. This condition, as discussed before, is not necessarily valid for real-world robots. However, this is a minor inconvenience since sidepayments are not allowed in Aumann's work. It is trivial, by comparison, to remove the superadditivity assumption instead of removing the TU assumption. He presents a proof that applying group rationality to the outcomes does not change the core, therefore, all "interesting" cores are equal. Thus, there is one core for any collective, regardless of the existence of TU. This makes much of the processing related to the core more straightforward, since there is a guarantee of only one core. This proof is presented in more detail in [4]. Aumann's work, coupled with a Bayesian core stability concept, provides a foundation from which additional uncertainty can be incorporated in a framework well suited to real-world robotic collectives. The introduction of uncertainty from all three sources, along with a formalization of CFDPs, forms the cornerstone of the research associated presented here. The uncertainty introduction and formalization methods are presented in the next chapter.

2.4 Coalition Formation Decision Processes

The distributed artificial intelligence (DAI) side of coalition formation focuses on the procedural aspects of coalition formation. It is concerned more with creating efficient and stable decision processes related to forming coalitions than the coalitions themselves. Algorithms for forming coalitions in polynomial time using greedy algorithms, ontology-based approaches, or reinforcement learning are applied by re-

[5]One often assumes that $H = v(S)$, which is a valid assumption for the real-world robotic domain.

searchers in DAI to generate coalitions that are "good enough", since finding the absolute best possible coalition structure takes O(N^N) time [35].

Shehory and Krauss [35] provide a greedy algorithm for forming coalitions. Their algorithm is distributed in that every agent in the collective simultaneously computes the coalitions. This is iterative:

1. Given knowledge of capabilities of each robot, for each task $t \in T$ where T is the set of all tasks,

   -compute the task values in the precedence set P_t.

   -If all tasks in P_t are assigned coalitions, set the value of all the tasks pV_t to the sum of the values of the tasks in P_t.

   -Otherwise, remove task t from T, since it cannot be performed.

2. Choose the task $t*$ with the maximum pV_t to be performed with its predecessors.
3. Remove $t*$ and its predecessors from T.
4. Update capability vectors of the associated members.

If a task t has predecessors, it implies a required ordering upon task execution, requiring completion of other tasks prior to executing t. Given this flow of information regarding the task, additional calculations are needed to determine the coalitions. Each agent A_i commits to calculating a subset of coalitions involving itself, communicating to extract capabilities and building a long-term commitment list L_i. Each agent A_i executes the algorithm to form a coalition as follows:

1. Locate in L_i the coalition C_j with the smallest cost c_j.
2. Announce this cost to all other agents.
3. Choose the lowest cost of all those that were announced. The corresponding coalition C_{low} and task t_{low} are also selected.
4. A_i joins the coalition C_{low} if it is a member of it.

5. Erase the task t_{low} from T corresponding to C_{low}.
6. Update capability vectors of all members of C_{low} according to contribution to the task.
7. Assign to L_i^{cr} the coalitions in L_i that need recalculations due to the members of C_{low} having changed their capabilities.

A variation where the agents that join a coalition are removed from further calculations until the coalition dissolves (upon task completion) is also presented. The total procedure has a computational complexity of $\mathrm{O}(n^k \cdot |T|^5)$, where n is the number of agents, k is the maximum size of a coalition, and $|T|$ is the number of tasks. The $|T|^5$ term comes from calculating each task's p-value pV_t taking $|T|^3$ time [35], performed up to $|T|$ times, for each of the tasks, giving $|T|^3 \cdot |T| \cdot |T| = |T|^5$. This algorithm is also very communication intensive, which often poses a problem in real-world robotic systems. This procedure, however, takes advantage of the distributed system, making each agent an expert in the coalitions it can possibly become a member of. Nothing is said explicitly by the authors about transferable utility (which is irrelevant to this approach) nor superadditivity, which is not applied. No stability concept is applied, with the exception of a formed consensus on the coalition formed by the algorithm. Given the nature of the algorithm and the constraints it places on the agents when forming a coalition structure, none need be applied. In other work [36], they present a variation that makes use of the Kernel stability concept. The primary difference between that and the algorithm presented here is that the agents check whether their calculations provide Kernel-stable coalitions and incorporates side payments to adjust the stability of potential coalitions.

Fanelli *et al.* [19] present a coalition formation decision process based on an ontology. The ontology they use breaks the robotic system into physical entities, functionalities, and interactions with other robots. The abilities of each robot in the multiple robot system (MRS) are inferred by examining each of these aspects. The functionality description is broken into actuation and sensorial capabilities. The

coalition formation mechanism evaluates links between motor schemas (as defined in [33]) as potential solutions. The procedure consists of two phases: an offline phase and an online phase. The offline phase connects robots of different types, defined by their sensorimotor capabilities through the ontology, and obtains the new capabilities of the proposed team. If at least one robot in the proposed team improves its ability to perform a task, the team is defined as a useful coalition. This procedure continues until all coalitions with no more members than a predefined limit are found. These results are used as inputs to the online portion of the procedure. The online phase applies a partial order to the tasks and determines a coalition utility. The coalition utility is defined as the mean of the utilities of the robots assigned to it. Robot i's utility U_i is determined using

$$U_i = w * \frac{\sum_{j=1}^{Q_i} q_{i,j}}{Q_i} - (1 - w) * \frac{\sum_{j=1}^{C_i} c_{i,j}}{C_i}$$

where w is a weighting factor relating quality factors ($q_{i,j} \in Q_i$) to costs ($c_{i,j} \in C_i$), based on functionalities. The algorithm iterates through robots, selecting available mates that can help it accomplish the current mission. It scans only the set of useful coalitions determined by the offline phase of the procedure. This is guaranteed optimal if there is a total order within competing missions. The guarantee of optimality is useful, especially since the total online processing is much reduced by comparison to the procedure presented by Shehory and Krauss. There are, however, a number of problems with this approach. First, the offline processing assumes three things: centralized processing, full knowledge of the capabilities of all other robots, and that these capabilities of robots do not change. If a robot happens to obtain a new capability (e.g., a new learned behavior), the offline process is invalidated. Additionally, if full knowledge of the capabilities of other robots is not present, the offline process is more likely to be invalidated by belief revision. Second, the algorithm makes use of all three potential sources of uncertainty. The robot capabilities are used to determine robot utilities, which are used to determine coalition utility. Uncertainty in

any one of the three sources could easily undermine the quality of the calculation and the ensuing coalitions that are formed. This approach, though effective with no uncertainty, is ineffective if uncertainty is incorporated. A more decentralized approach would contribute to more robustness in the face of uncertainty.

Abdallah and Lesser [1] make use of a Q-learning approach applied to a neural network as a means to prune a search tree. Their formal problem representation defines $T = \langle T_1, T_2, ..., T_q \rangle$ as the sequence of tasks arriving in an episode. An episode defined as a time segment where agents receive task sequences and are assigned to tasks. Each task T_i is defined as a tuple $\langle u_i, rr_{i,1}, rr_{i,2}, ..., rr_{i,m} \rangle$ where u_i is the utility gained if task T_i is accomplished (the reward) and $rr_{i,k}$ is the amount of resource k required by the agents to complete the task. Each agent $I_i \in I = \{I_1, I_2, ..., I_n\}$ controls some amount $cr_{i,k}$ of resource k. Each agent I_i is therefore defined as the tuple $\langle cr_{i,1}, cr_{i,2}, ..., cr_{i,m} \rangle$. The coalition formation problem is then defined as finding a subset of tasks $S \subseteq T$ that maximizes utility. Mathematically,

$$\sum_{i|T_i \in S} u_i \geq \sum_{i|T_i \in S} u_k \quad \forall k \neq i. \tag{2.1}$$

Let $C = \{C_1, C_2, ..., C_{|S|}\}$ be defined as a set of coalitions. The relationship in Equation 2.1 holds when there exists C where $C_i \subseteq I$ is the coalition assigned to task T_i such that

$$\forall T_i \in S, \forall k : \sum_{I_j \in C_i} cr_{j,k} \geq rr_{i,k} \text{ and } \forall i \neq j : C_i \cap C_j = \emptyset.$$

In other words, each task has at least enough resources allocated to it that it can be completed, and the coalitions are disjoint (no agent is a member of multiple coalitions). Note that the coalition formation problem defined here makes no effort to minimize resource waste, which occurs if $\sum_{I_j \in C_i} cr_{j,k} > rr_{i,k}$.

The approach to coalition formation that Abdallah and Lesser present applies a hierarchy to the agents in the collective. This makes each agent into a local expert regarding its sub-hierarchy, in that they know the resources that are available to every

agent it manages. A task can be discovered by any agent, which queries the resources in its local hierarchy and, if the task can be accomplished, selects the best child in the hierarchy to execute the task. If the resources are not available, the task moves up to the next higher level in the hierarchy and the manager at that level selects the best child. A child can be either an individual agent or a lower-level manager, which itself has multiple children. This hierarchy provides a centralization of the coalition formation process and allows for coalition "templates", since the formed coalitions are subsets of the organizations in the hierarchy. The learning is applied to the order in which the queries are made to the children, streamlining the task assignment over time. The result is an approach to coalition formation that scales well and outperforms a traditional greedy selection approach. However, the organization must be hand-coded, preventing dynamic reconfigurability. The approach is also somewhat centralized, since failed tasks travel upwards in the hierarchy to the topmost organization manager. Finally, the knowledge possessed by the managers must include the resources available to every member of its organization (agent type) and reason about its contribution (agent utility) to determine a coalition that fulfills a total utility (coalition utility). This, therefore, requires full knowledge of all three sources of uncertainty. Incorporation of uncertainty into this approach would affect the entire collective adversely. The formal definition of the problem space that they present does allow for other approaches, and many algorithms would fit well into this framework.

2.5 Multi-robot Task Allocation

There is difficulty associated with transferring the DAI approaches into multi-robot systems, as the multi-robot systems are subject to real world constraints [41].

There are many approaches to multi-robot task allocation, including ALLIANCE [33], ASyMTRe-D [39], MURDOCH [20], Dynamic Role Assignment [10], Broadcast of Local Eligibility [43], and First-price auctions [46]. These systems all make an assumption that each task can be completed by a single robot, or the number of robots for a task required is known *a priori*.

ALLIANCE [33] uses *impatience* and *acquiescence* to define a robot allocation to a task in a decentralized manner. The tasks are assigned to the entire collective, and one robot is selected to execute it. If it fails to execute the task in a certain time, another robot becomes impatient and attempts to take over. If the original robot is sufficiently acquiescent, it allows the new robot to attempt the task. This provides the ability for dynamic allocation of robots to a task, but each task is inherently a single robot task. In essence, ALLIANCE allows robots to decide among themselves which robot attempts a task. No provisions are made for multi-robot tasks (those tasks which require multiple robots to execute), though it is a straightforward variant to limit the acquiescence of an executing robot yet allow a new robot to join. The biggest limitation of ALLIANCE is that there is no self-regulation which prevents allocation of the entire collective to a task. The collective can then be stuck trying to execute an impossible task. Preallocating multiple robots to an inherently multirobot task, as in coalition formation, constrains the extent to which the robots dedicate resources to an impossible task.

ASyMTRe-D [39] uses a distributed version of the "schema" abstraction [3] to generate coalitions. The approach applies a distributed negotiation process based on the Contract Net Protocol [37]. ASyMTRe-D creates motor schemas and defines them as resources available to other robots. This allows the robots to solve for the resources required for a task and gather those resources. This naturally creates a coalition for the task's execution. Provided each robot in the collective understands the schemas available and the contributions of those schemas for task execution, the robots can build successful and robust coalitions. However, ASyMTRe-D is built assuming that no new robot types join the collective, and those in the collective understand *a priori* how to represent and use every schema required for a task. ASyMTRe-D does allow robots to join or leave the collective frequently, but any robots joining must be well understood (i.e., the schemas are available to every other robot and every other robot understands the information needed to activate the schemas) ahead of time. This

prior knowledge about other robots precludes ASyMTRe-D's applicability to more dynamic collectives.

MURDOCH [20] allocates tasks to robots with a first-price auction method [31]. It announces a task with defined metrics, then the robots issue bids. The task issuer (auctioneer) awards the task to the highest bid then monitors task completion. This requires that the tasks are inherently single robot tasks, assigned to a collective of multiple robots. Under this method, only one robot is assigned to a task and no provisions are made to allow multiple assignment. This approach is limited to tasks that require a single robot unless a multirobot task is explicitly decomposed into distinct single-robot roles. This requires extensive knowledge on behalf of the auctioneer to decompose the task to single robot roles. Without significant user intervention, this requires a static collective.

Dynamic Role Assignment [10] provides a method for task *execution* where the robots can dynamically change roles. The roles are defined ahead of time, but the robots assigned to each role may change dynamically. This enables a multi-robot collective to readjust the allocation of tasks during execution, making for a more efficient execution. One drawback of this system is that the roles must be defined by a user. Thus, each task has a representation that describes the specific steps and roles to fill for successful execution. This approach is not well suited to an automated role generation, since the robots would require full knowledge of the other robots in the collective to determine the plan and the roles. Furthermore, the task is already allocated when Dynamic Role Assignment takes over, so coalition formation and task allocation discussions are impertinent.

Broadcast of Local Eligibility [43] uses the behavior-based architecture paradigm [9] in a multi-robot setting to allocate tasks to the collective. It allows identical robots with identical behaviors (called peer behaviors) the ability to cross-inhibit each other. The robot assigned to the task is the most eligible robot for that task. A notable drawback of this representation includes the underlying constraint that the robots are

identical. One way to reduce this constraint is to allow heterogeneous hardware with identical behaviors, but the issue then is in matching the functionality, parameters, and feedback such that the behavior output is identical. Another issue is that the tasks are inherently single robot tasks, assigned to a collective of multiple robots. Each task assigned to the collective is executed by a single robot, and the only contributors to comparative eligibility (quality for task execution) are physical location in the environment and current tasking.

The focus of auction methods is to address efficiency and optimality [17, 20, 21] in multirobot task allocation. They operate by allowing robots to bid on a task, and, much like an auction, awarding the highest bidder with the task. They operate well in the face of uncertainty, but make a number of assumptions which limit their applicability. First, they assume that the robots are generally not selfish. Since there are times when a robot must be selfish (for example, when a robot's battery charge level is critically low), this assumption does not hold for all cases. Secondly, the existence of an auctioneer centralizes the approaches. The centralization can be mitigated by allowing any robot to become an auctioneer, but the centralization is present for each task nevertheless. The third, and most critical, assumption that auction methods make is that the number of robots required for a task are known *a priori*. In circumstances where the procedural aspects of tasks are not well defined, this assumption may prevent successful execution. For example, a task requiring box pushing (such as in [42]) may assume one agent is required. When that agent is incapable of pushing the box alone, the task must be reassigned appropriately. Therefore, though the utility generation and evaluation aspects of the procedure to be presented in Chapter III are similar to aspects of auction methods, the procedure followed uses agent preferences instead of a predefined coalition size for determining the size of the optimal coalition.

2.6 POMDPs

Some approaches to planning under uncertainty have been through extension of Partially Observable Markov Decision Processes (POMDPs) [28]. The POMDP of agent i is defined as

$$POMDP_i = \langle S, A_i, T_i, \Omega_i, O_i, R_i \rangle$$

where:

- S is a finite set of states for the environment
- A_i is a set of actions for agent i
- T_i is a set that defines the probabilities of arriving in certain states if agent i takes action a at another state
- Ω_i is agent i's observation
- O_i is agent i's observation function: it defines probabilities of observations given an action
- R_i is agent i's reward function, representing i's preferences

This considers only a single agent. The extensions expand this representation to multiple agents. The first extension is the Decentralized POMDP (DEC-POMDP) [6]. The DEC-POMDP is defined similarly to the POMDP, except the transition probabilities (T_i), observation (O_i), observation function (Ω_i), and reward function (R_i) become dependent upon the joint actions $\langle a_1, a_2, ..., a_n \rangle$. The DEC-POMDP adds the benefit of multiple agents, but assumes that all agents are fully cooperative. That is, they have a common reward function and form a team. In fully cooperative domains, the best team to form is always the grand coalition, which is impractical for the application under consideration here.

Another POMDP variant is the Interactive POMDP (I-POMDP) [24]. This framework represents the state as a set of interactive states which are dependent upon an agent's internal model of the other agents. These models represent the

full type of an agent, including belief states and optimality criteria. The beliefs of agents are potentially infinitely nested: an agent's beliefs about another agent's beliefs may include beliefs about the first agent, etc. This variant is applicable to coalition formation for real world applications, with one significant exception. The already time consuming approaches to coalition formation must be injected into the I-POMDP representation, and the processing over the interactive states and beliefs inherent in the I-POMDP must also be performed. This provides a model that, while applicable, is more cumbersome than necessary for coalition formation applications. The models of agents in an I-POMDP are also significantly more complex than in the framework presented in Chapter III, making even more work for each agent whenever an update occurs.

2.7 Summary

The game theory approaches to coalition formation presented in Section 2.1 use rational behavior to define stability in coalitions. These approaches rarely translate seamlessly to the real world. For example, the example of the core stability concept indicated that, rationally, all producers of A take all the profit, and all producers of B take none of it. In the real world, very few producers would join a coalition without some personal profit. These concepts do, however, provide solid rational allocation schemes and stability approaches and are applicable to many systems nonetheless.

Section 2.2 presents approaches that handle uncertainty and a discussion of the uncertainty sources themselves. When an agent's motivations, hardware construction and configuration, and general capabilities are uncertain, agent type uncertainty is present. One approach that directly addressed agent type uncertainty incorporates a modified core, called the Bayesian core, presented by Chalkiadakis and Boutilier [11]. Agent cost (or agent utility) uncertainty occurs when the expense of executing a task–recorded by energy consumption, communication channel use, or computation time–is uncertain. This was addressed in an indirect manner by Kargin [29]. The third source of uncertainty is coalition value. In environments where NTU is present, the value

of the coalition can be affected by inaccurate state representations. Blankenburg *et al.* [7] address this source of uncertainty. Section 2.2.4 completes the discussion of the uncertainty sources by addressing the relationship between the three sources.

Due to the importance of NTU to the domain in which the proposed solution is applied, it is addressed directly in Section 2.3. Aumann [5] presented a proof that there are many parallels between TU and NTU systems, and the stability may be evaluated in a similar manner. This leads to an evaluation of the DAI side of coalition formation, which is concerned with the mechanisms applied to form coalitions. This is presented in Section 2.4. The approaches presented in that section do not incorporate uncertainty, but serve instead to indicate the breadth of approaches that can lead to coalitions. The formalization of the problems and the domains are widely different from one approach to another. The disparate representations reflect very little continuity and consensus in the DAI aspects of coalition formation. Section 2.5 provides the multi-robot task allocation perspective. These approaches apply various tools to assign a set number of robots to a task. Since this number is not always available *a priori*, the applicability is limited to static collectives. Section 2.6 discusses the POMDP and variants as a modeling technique but the multiagent applicability of the POMDP is limited. An alternate POMDP called the I-POMDP is applicable to multiagent systems and applicable to coalition formation, however, due to the complexity of the model and long processing times, is unnecessarily cumbersome. The applicability is not diminished, however the practicality of applying I-POMDPs is limited due to these issues. Chapter III presents a formalization of the coalition formation decision process that is more widely applicable. Additionally, an approach is proposed that addresses uncertainty in all three sources directly. An algorithm is developed that allows for this uncertainty.

III. Methodology

Uncertainty is not well addressed in current coalition formation approaches. The approaches that do address uncertainty handle it in such a limited manner as to make the mechanism cumbersome, rather than flexible and more generally applicable. This precludes application to heterogeneous, decentralized systems of autonomous mobile robots. This chapter refines the Coalition Formation Decision Process (CFDP) model to incorporate a procedure designed for heterogeneous and general collectives through a representation which includes provisions for uncertainty.

The uncertainty is incorporated into the agent model where each of the three sources of uncertainty (agent type, agent cost, and coalition value) are addressed. The procedure incorporates the expectations associated with the uncertainties and performs processing on these expectations to generate stable coalitions that hold with the strict Bayesian core [14][1]. The ensuing coalitions are robust, and a member of a coalition can multitask by joining multiple coalitions, since a coalition only addresses a single task. The manner in which the other agents are modeled has a significant effect on the potential coalitions, and the mechanism allows for an agent-specific, not system wide, configuration. Improved values in the state vector and covariance matrix that represents the relationship between agents are achieved by repeated interaction of agents in coalitions and the formation processes. The formal representation presented in Section 3.1 is structured to enable the incorporation of uncertainty. The solution in Section 3.6 uses this representation to form robust coalitions.

3.1 The Coalition Formation Decision Process

Let $N = \{1, 2, 3..., n\}$ be the set of agents in a collective. The set of all possible coalitions is Ω (this includes the empty set, so Ω is the powerset of N, and its size $g = 2^n$). The coalition structure, $CS \subseteq \Omega$, is the set of all coalitions formed. The mth coalition that can be formed in this collective is $S_m \subseteq N$. Intuitively, $S_1 \cup S_2 \cup ... \cup S_g =$

[1]The two key differences between the strict Bayesian core and our variant on the core are that we allow agents to leave a coalition as desired, and there is no transferable utility. Otherwise, the coalitions that form do hold with the strict Bayesian core.

N. This framework relaxes the single coalition limitation on an agent imposed by much of the previous work. An agent can join multiple coalitions at once, so it is not necessarily true that $S_m \cap S_\mu = \emptyset \quad \forall m \neq \mu$. The tasks assigned the collective are $T = \{T_1, T_2, ..., T_p\}$, where p is the total number of tasks. Each coalition $S_m \in CS$ has an associated task $T_e \in T$. Note that $m = e$ or $m \neq e$ is irrelevant, since the sequencing of the coalitions in S and the tasks in T are arbitrary. Each task T_e may have a sequence of subtasks, defined as $\langle t_{1,e}, t_{2,e}, ..., t_{r,e} \rangle$, where r is the index of the last subtask in the task sequence. Each task also has a payoff U_e. The task cost C_e is equal to the coalition cost $x_{T_e}^{S_m}$, which is one source of uncertainty. Thus, a task T_e is a tuple, $\langle U_e, t_{1,e}, t_{2,e}, ..., t_{r,e} \rangle$. The cost for the task is not included in the tuple since the cost is dependent upon the coalition assigned to the task.

The CFDP is defined through a tuple. Formally, an agent is represented by the tuple $\langle N, T, V_i, M_i, Q_t^i, C_t^i \rangle$ where:

- N is the set of all agents, thus ($CS \subseteq \Omega$), where $S \in \Omega$
- T is the set of tasks where $t_1, t_2, ...t_p \in T$
- M_i is agent i's set of models regarding the other agents' abilities
- Q_t^i is the quality value for task T for agent i. This is how "good" agent i determines itself to be at task T on (0,1].
- C_t^i is the set of costs for task T that agent i expects for the other agents
- $V_t^i(S, t, M_i, C_t^i)$ is the utility for agent i for task t given C_t^i, Q_t^i, and M_i

The optimal coalition for task t is the coalition $S \subseteq N$ that maximizes $U_t^i \ \forall i \in S$, and, for all coalitions that match this requirement, $\sum_{i \in S} U_t^i \geq \sum_{i \in R} U_t^i$ for $R \neq S$. In other words, no agent i in S can reasonably expect to form a coalition for task t where U_t^i is greater than U_t^i in S, and the total profit is higher for S than for any other coalition. A specific agent in S may not receive its maximum possible payoff, but the payoff is the maximum *achievable* payoff since the other agents in the coalition providing its maximum payoff may not desire to form it. S is not necessarily unique.

Also, the tasks are solved for individually and each task has a single coalition, so this representation is greedy in terms of assigning coalitions to tasks.

The representation of the CFDP as a tuple reflects the four parameters that affect the coalition formation: The task, an agent's models, an agent's quality, and an agent's cost. The effects of these parameters are partially investigated in Chapter IV.

3.2 Agent Model

The stability approach for this work, a variant of the core [23], is defined as the set of coalitions where no agent can improve its payoff. More specifically, the applied version of the core is a slightly modified strict Bayesian core [14]. The strict Bayesian core applies to collectives where all the agents believe they cannot obtain a better payoff by joining another coalition. The strict Bayesian core requires knowledge of agent payoffs for each potential coalition. The agent payoff is needed so that each agent can reason about the other agents and determine whether a given coalition structure is in the core. The agent type is necessary to calculate the total agent cost in a coalition, which, in turn, dictates this agent's payoff.

The agent model is the main component allowing for uncertainty. It also enables a fully decentralized system to handle arbitrary agent addition or removal. Define agent i's "type" as $A_i = \langle a_1^i, a_2^i, a_3^i, ..., a_f^i \rangle$, where each a_τ^i is an ability. This collection of abilities (each composed of ability type, quality, and standard deviation) which describe an agent's estimates about another agent's task execution quality is called an *ability set*. Capturing an agent's type in this way allows other agents to represent an agent by what it can do, not what it is. This provides greater flexibility, since the representation of what a robot "is" may be difficult or impossible for another robot to capture appropriately. Robots can represent their own abilities internally, and it is this internal representation that another robot's type is translated to. More specifically, A_i as presented above is the true robot type, represented internally by

robot i. Another robot j characterizing h abilities represents i as

$$A_i^j = \{a_1^{ij}, a_2^{ij}, a_3^{ij}, ..., a_h^{ij}\}.$$

Through this characterization, each robot models the other robot types with these ability sets, which enables transforming another robot's type into its own, well understood space. An aspect of this representation is that the set of abilities robot j uses to define i might not accurately represent i, but the evaluation of the abilities can yield outcomes similar or identical to a perfect model of robot i. In other words, the models can be similar or identical in *result* without being similar in composition.

Define the quality of an ability a_τ^{ij} as $q_\tau^{ij} \in (0, 1]$. The quality values are used to determine a robot's estimate of how good another robot is at a task. To make this determination and enable modification of the values, these qualities are arranged to define a vector state for a modified Kalman filter:

$$\hat{x^{ij}} = \begin{bmatrix} q_{t1}^{ij} \\ q_{t2}^{ij} \\ q_{t3}^{ij} \\ \vdots \\ q_{th}^{ij} \end{bmatrix} \tag{3.1}$$

The standard deviations for each ability, $\sigma_{a_1^{ij}}, ..., \sigma_{a_h^{ij}}$, are used (in part) to define a covariance matrix $P_{h\times h}^{ij}$. To define this, the relationship between the abilities are assumed known (i.e., each robot understands how changes in one of its own ability qualities affects other abilities). Each agent captures all of the agents in the collective in this way, including itself (except the covariance is an $h \times h$ matrix of 0's for itself). This makes the reasoning about potential coalitions more straightforward. As an agent negotiates tasks in coalitions, updates on the quality and covariance values are performed through the modified Kalman filter.

3.2.1 Kalman Filter Representation. A modified Kalman filter is used for the evaluation, estimation, and update of the ability values. The values improve over time with Kalman filters, thus making for a better estimate of other robots' abilities and improved coalitions formed from the estimates. The updates for agent type occurs at two times during task execution: at the beginning of the coalition formation procedure and after task completion. The update performed at the beginning of the procedure is based upon quality values, which all agents broadcast at the beginning of the negotiation. Immediately after receiving another agent's quality, each agent determines the ability sets it expects the other agents to use for the task under consideration. These abilities are used to generate an expected quality, and the residual is used to generate an update. When an agent broadcasts its own quality, the other agents reconcile this value with the quality they expect. At the end of task execution, the agents determine the cost each agent truly incurred and use this cost to infer quality, again reconciling this value with the quality they expected. As with a general Kalman filter, our modification has two stages: propagation and update.

The update stage incorporates measurements, which in this case are the qualities provided by the other agents. The update step for a general Kalman filter is

$$\hat{x}^{ij}_{k+1^-} = \mathbf{F}^{ij}\hat{x}^{ij}_{k^+} + \mathbf{B}^{ij}u^{ij}_k + \mathbf{G}^{ij}w^{ij}_k. \tag{3.2}$$

Where $\hat{x}^{ij}_{k+1^-}$ is the state at time $k+1$, before the update at time $k+1$. The state at time k, after update is $\hat{x}^{ij}_{k^+}$. The input at time k is u^{ij}_k, and w^{ij}_k is the noise at time k. The state transition model is the matrix $\mathbf{F}^{ij}$, $\mathbf{B}^{ij}$ is the control input matrix, and $\mathbf{G}^{ij}$ is the noise model.

The covariance in a Kalman filter is updated in a similar manner, using

$$\mathbf{P}^{ij}_{k+1^-} = \mathbf{F}^{ij}\mathbf{P}^{ij}_{k^+}(\mathbf{F}^{ij})^T + \mathbf{Q}^{ij}_k \tag{3.3}$$

where $\mathbf{P}^{ij}_{k+1^-}$ is the covariance at time $k+1$ before update (and is positive, semi-definite), $\mathbf{P}^{ij}_{k^+}$ is the covariance at time k after update (also positive, semi-definite), and $\mathbf{Q}^{ij}_k$ is the covariance of w^{ij}_k. Since the ability qualities have no true dynamics and there is no reason to expect drift during the propagation stage, let $u^{ij}_k = 0,\ w^{ij}_k = 0$ $F^{ij} = I_{n\times n}$, and the process noise is nonexistent ($\mathbf{G}^{ij} = 0$). Therefore $Q^{ij}_k = 0$. In other words, the propagation stage merely passes the previous value, so the state and covariance propagation steps from Equations 3.2 and 3.3 become

$$\hat{x}^{ij}_{k+1^-} = \hat{x}^{ij}_{k^+} \tag{3.4}$$

and

$$\mathbf{P}^{ij}_{k+1^-} = \mathbf{P}^{ij}_{k^+} \tag{3.5}$$

respectively. This means that the propagation from post update at time k to pre-update at time $k+1$ is a simple carryover of the value.

The update step requires the definition of an observation model and the measurement noise. When a task is provided, the agents determine an ability set

$$\{a^{ij}_{T,1}, a^{ij}_{T,2}, a^{ij}_{T,3}, ..., a^{ij}_{T,\gamma}\} = A^j_i(T) \subseteq A^j_i$$

that represents the abilities that j expects i to use to execute task T. This is translated to an observation model $\mathbf{H}^{ij}_{1\times h}$, where the indices of each $A^j_i(T)$ are marked with $\frac{1}{\gamma}$ and the others marked with 0. These values in $\mathbf{H}^{ij}$ define how much insight an agent has regarding the contributions of each ability to the observation. For example, if $h = 3$ and the first and third abilities are in $A^j_i(T)$, then $\mathbf{H}^{ij} = \begin{bmatrix} 0.5 & 0 & 0.5 \end{bmatrix}$.

Since the abilities an agent can represent do not necessarily reflect the true abilities, there is measurement noise inherent in the system. The measurement z_k is

$$z^{ij}_k = \mathbf{H}^{ij}\hat{x}^{ij}_k + v^{ij}_k$$

where v_k^{ij} is the measurement noise. Since the agents receive measurements as communicated messages from other agents, the measurement z_k^{ij} is the quality received from the other agents, which occurs at the beginning of the algorithm. The scalar z_k^{ij} is such that $\mathbf{H}^{ij}$ projects the $h \times 1$ state into the scalar measurement space.

The filter's update is performed by determining the residual

$$y_k^{ij} = z_k^{ij} - \mathbf{H}^{ij}\hat{x}_{k^-}^{ij}$$

and the residual covariance

$$\mathbf{M}_k^{ij} = \mathbf{H}^{ij}\mathbf{P}_{k^-}^{ij}(\mathbf{H}^{ij})^T + \mathbf{R}_k^{ij}$$

where $\mathbf{R}$ is the covariance of v_k. The measurement noise v_k is assumed zero mean white Gaussian noise. Its variance is user-defined, and the variance is the value for R. The Kalman gain is determined using

$$\mathbf{K}_k^{ij} = \mathbf{P}_{k^-}^{ij}(\mathbf{H}^{ij})^T(\mathbf{M}_k^{ij})^{-1}$$

where $\mathbf{M}$ is invertible. This gain is applied to the residual and the previous state to determine the post-update state using

$$\hat{x}_{k^+}^{ij} = \hat{x}_{k^-}^{ij} + \mathbf{K}_k^{ij}\tilde{y}_k^{ij}.$$

Finally, the post-update covariance is determined using

$$\mathbf{P}_{k^+}^{ij} = (I - \mathbf{K}_k^{ij}\mathbf{H}^{ij})\mathbf{P}_{k^-}^{ij}. \tag{3.6}$$

This cycle is repeated each time the agents begin negotiation on a task to improve the estimates of the ability qualities over time.

This model and filter combination provide the system with the ability to handle dynamic addition or removal of robots from the collective. When a robot is added, a new model (the entire Kalman filter construction) is built and initialized, with the ability set and the covariance matrix constructed by each agent already in the collective to model this agent. The abilities to be used for a task are generated by the modeling agent's control architecture's internal planner and these abilities are considered "observable". Over repeated negotiations for the same task, a robot's model is modified to more closely match the value which another robot sent for its quality. If an agent expects that previously unused abilities will be used for a given task, the internal structure of the filter enables adjustment of the previously unused abilities and fine-tuning of the abilities which had been used before. In this way, a robot's internal representation of another robot can converge and generate the same result as a perfect model, without being of the same composition. In other words, this approach allows a robot to generate an accurate quality for another robot while having an inaccurate model.

The quality values provide little insight into the internal structure and valuation for the agents' true abilities. However, the residuals between the provided utility and the expected utility do provide some insight. If the residual is small, the adjustment from the previous estimates for the expected abilities to the new estimates is small. Another factor that affects the rate at which the models are adjusted is the covariance matrix. When the covariance matrix contains small values, the values adjust little, regardless of the residual value. Over time, the covariance matrix values also diminish, so repeated encounters of tasks using the same ability increases the agent's estimate of the ability's accuracy.

In summary, the agent modeling technique allows great flexibility in the collective, requiring only that an agent initialize a new ability set vector whenever a new agent is added to the collective. The subset of the abilities used for a task are solved for using an internal planner built into the agent's control architecture. This planner may be static or integrated in such a way as to generate ability sets for a task. The

abilities selected for the execution of the task determine the positions of the nonzero values in the **H** measurement model, and the number of abilities in the subset dictate the values in those positions. The user-defined values of intial covariances on abilities and **R** determine how quickly an agent's model converges onto a given quality value and how resistant it is to changing the quality values, since the covariances and **R** define post-measurement covariances.

3.3 Agent Cost

Agent cost is the resource expenditure an agent experiences while executing a task. Two agents of different hardware construction following the exact same path may incur different costs. The task type helps to refine estimated resource expense, but is not guaranteed to improve the estimate. The agent type A_i dictates the agent's ability to perform a task, though not the cost it encounters for the task execution.

The agent cost is a function that determines the estimate based on the task, the agent's state model, and the agent type $c_i = f(T_e, S_m, A_i)$. For example, consider a task where a single agent is required to move to a specific position. If this position is accessible through a straight line, the resource expense is readily inferred using the vector length as a basis. If the position is behind a wall or otherwise obstructed, the path to the position is necessarily longer than the direct vector's length, increasing the cost. However, assuming the domain is fairly well known–paths to target positions can be developed *a priori*, etc.–a resource expenditure estimate provides a reasonable basis for calculation of total cost. Given a poorly defined environment, these estimates are the best information available to an agent for its representation and processing.

The cost c_i is determined by breaking down T_e into its sequence of steps, $\langle t_{1,e}, t_{2,e}, ..., t_{r,e} \rangle$ and determining the costs associated with each step. This is determined from an agent's subjective understanding of the environment and estimates of its resource expense related to a task. This gives an estimated cost for each subtask, $\langle c(t_{1,e}), c(t_{2,e}), ..., c(t_{r,e}) \rangle$. Since a plan with abilities needed for each subtask is

available from the internal planner, the cost c_i^j is determined using

$$c_i^j = \frac{1}{\gamma} \sum_{t_{b,e} \in T_e} \frac{c(t_{b,e})}{q_1^{ij}} + \frac{c(t_{b,e})}{q_2^{ij}} + ... + \frac{c(t_{b,e})}{q_\gamma^{ij}}. \tag{3.7}$$

Equation 3.7 indicates that each ability scales the estimated cost for the task, generating an ability-centric cost. The ability-centric costs are added to each other for each subtask, creating a subtask cost. These subtask costs are scaled by the number of abilities to avoid penalizing agents with more abilities. The subtask costs are summed to create a total cost, c_i^j, or the total cost that agent j expects agent i to incur. With this relationship, the abilities with lower qualities needed to execute a task increase cost more significantly than abilities with higher qualities (note that the abilities are scaled to between 0 and 1). The cost estimates can be learned over time if the domain is not well known, or generated ahead of time if it is well known. In general, it is assumed that the domain is not well known and the costs are learned over time.

3.4 Coalition Cost

The coalition cost should be minimized in order to maximize net payoff, which is distributed to the players. This minimization allows for agents to determine their best coalition, where their net payoff as a part of the whole is as high as possible, according to their estimates. The coalition with the highest net profit is the optimal coalition, as defined for this application. The estimates improve over time with a Kalman filter update, where the task cost c_i^j and the estimates of the types of other agents A_i^j are adjusted based upon proposed coalitions and true execution costs.

The interconnectedness between the three sources of uncertainty is important, since it captures the coalition's capabilities and costs appropriately. The coalition cost $C_{S_m}^j$ for coalition S_m is the sum of the agent costs in the coalition, which is based

upon the agent types:

$$C^j_{S_m} = \sum_{i \in S_m} c^j_i = \sum_{i \in S_m} \frac{1}{\gamma} \sum_{t_{b,e} \in T_e} \frac{c(t_{b,e})}{q^{ij}_1} + \frac{c(t_{b,e})}{q^{ij}_2} + ... + \frac{c(t_{b,e})}{q^{ij}_\gamma}. \tag{3.8}$$

This cost is applied to the coalition formation algorithm for generation of a profit vector $D^j_S = \{d^j_1, d^j_2, ..., d^j_\eta\}$ of a given coalition S_m where

$$d^j_i = (U_T - C^j_{S_m}) \cdot \frac{1 - \frac{c^j_i}{C^j_{S_m}}}{|S| - 1},$$

which is the estimated profit for agent $i \in S$ according to agent j. The sum of the profit vector in relationship to the payoff of the task is the coalition value. The coalition value is not calculated explicitly, as it is implied through this relationship. The coalition value is based entirely on agent cost, which itself depends on the environment and agent types. The agents improve the quality of their agent type representation using a variant of a Kalman Filter (Section 3.2.1). This allows each agent to represent the other agents in their own reference frame.

3.5 Model Relationship Discussion

Distinctly different models do not affect the number of communication messages. For example, one agent may propose a coalition with profit vector outside the tolerance range of one of the potential members. Since this coalition is proposed, it is naturally within the tolerance range of the proposer. An agent whose acceptable range is outside of this tolerance declines the coalition since the tolerance ranges are so different. This causes an agent to potentially pass on coalitions where it has been offered more profit. In these cases, the disagreeing agent expects that it would not contribute as much to the coalition as the proposer agent expects and therefore, by declining the proposal, avoids some of the issues that may occur during execution associated with model differences. Over time, these agents may have a convergence in their models which allows them to propose acceptable coalitions to each other. Additionally, an agent

may end up losing out on a portion of the profit or gaining more profit if it does not have the lowest timestamp. Each message (or group of messages, in a broadcast) is sent with a specific timestamp. In race conditions, interference, or any other sort of communication-based impasse, the lowest timestamp is the winner. Proposals of the same coalition are given to the agent who has assigned the lowest timestamp to their message, regardless of whether this proposal is within the tolerance range of the potential members. If they are within tolerances of the coalition members, each member accepts the distribution as proposed, even if it means an increased or decreased profit for a non-proposer, relative to its own model. Therefore, the models must have fairly similar results (but not necessarily structures) to form a coalition, since any member with distinctly different results from their model disagrees with a proposal and the coalition does not form. This occasionally prevents the formation of viable coalitions, especially in the early stages of implementation. Given these, any model disagreement does not affect communication volume, but it does affect the results of the communication, such as reaching an agreement to form a coalition. The procedures followed to form coalitions enforce a degree of model agreement to a point where the results of reasoning across an agent's model of another agent must provide a similar result to the agent's true model, within the tolerance range for acceptance. This approach still holds with the strong Bayesian core presented by Chalkiadakis and Boutilier [14], since each agent takes the best profit according to its estimates. Part of its estimates in this framework is skepticism about the disparate nature of the offered coalition and its understanding of an acceptable coalition of identical agent composition. Though the agent may be told it can receive the best profit in a certain coalition, its estimates prevent it from joining the coalition as proposed since it does not expect that the coalition can make good on the offered profit.

A benefit of this representation is that the system is more robust to interference from a rogue agent. The rogue agent may propose coalitions that offer most of the profit to the other members in order to skew the formation or force tasks to fail. Agents which are members of these proposed coalitions will decline the proposals, even if there

is a significant potential gain. The model agreement enforces the practical view of things: if something sounds too good to be true, it probably is. Model differences may prevent formation of a coalition to execute a task, but this implies that the members of the collective either expect the task is impossible or they do not have enough information about each other to form reasonable coalitions. The Kalman filter updates occurring at the beginning of the coalition formation procedure allow the models to converge over time, so the same task presented multiple times may eventually become executable.

3.6 Algorithm

One of the challenges faced by the application of coalition formation to complex domains and robotic systems is the limitations imposed by the control architecture the systems use. Often, the coalitions are static and their tasks are limited to well-defined problems. This limits the dynamics of the collective. Certain control architectures are designed for loosely coupled tasks and, therefore, have more loosely coupled robots, allowing for greater dynamics in terms of coalition formation. ALLIANCE [33] is one example. A recently developed control architecture called HAMR [26] maintains loose coupling for the robots while still allowing tightly coupled tasks. The mechanism that provides this is based on utility values. Each robot produces an estimate of its quality for a given task. This estimate is based on the robot's perfect knowledge of its abilities and estimates of resource expense. The quality value is broadcast to all members of the collective so they can individually process the task allocation. A coalition formation mechanism can take advantage of these quality values to gain insight into the broadcasting robot's abilities and form improved coalitions. However, since the abilities a receiving robot can represent may not be the same as the broadcaster, the abilities are captured through uncertainty.

The procedure followed to generate proposed coalitions assumes unreliable communications to an extent that the communication messages are received intact, but may take multiple communication attempts. The coalition formation decision proce-

dure is broken into two roles: proposer and responder. These roles are concurrently executed in each agent. The proposer generates coalitions based upon the agent's subjective view and suggests them to other agents. The responder receives a suggested coalition and determines the quality of the proposal given the agent's subjective view of the potential coalition. Upon receipt of a proposed coalition, the agent no longer needs to propose that coalition to the other agents, so the responder role reduces the workload of the proposer. The proposer's procedure for agent i is shown in Algorithm 1.

Algorithm 1 Proposer(Task T)

1: Determine and broadcast quality and subtask costs
2: Receive quality from other agents. Update Kalman filter and adjust ability quality values.
3: **for** each $S \subseteq N$ where $i \in S$ **do**
4: **for** each agent j **do**
5: generate required abilities A_j^i
6: **end for**
7: determine $C_S = \sum_{j \in S} c_j^i$
8: $\forall j \in S$, calculate $d_j^i = (U_T - C_S^i) \cdot \frac{1 - \frac{c_j^i}{C_S^i}}{|S|-1}$, save as D_S^i
9: **if** $d_j^i : j = i < d_i^i$ *singleton* **then**
10: break
11: **end if**
12: **for** each $j \neq i \in S$ **do**
13: send S, D_S^i to j
14: **if** response = negative **then**
15: break.
16: **end if**
17: **end for**
18: save S in PS (the set of acceptable coalitions)
19: **end for**
20: **while** a coalition is not formed and $PS \neq \emptyset$ **do**
21: select from PS the S_m maximizing d_i^i.
22: issue S_m to its members of S_m for final approval.
23: **if** approved **then**
24: announce S to collective and exit.
25: **else**
26: remove S from PS.
27: **end if**
28: **end while**

Algorithm 2 Responder(message)

if message type is final approval with S, D_S^j **then**
  if i is a member of an approved coalition W **then**
    let $d_i^j \in D_S^j$ and $d_i^l \in D_S^l$
    if $d_i^j < d_i^l$ **then**
      respond with negative
    else
      respond with positive, notify proposer of W
    end if
  end if
else
  receive S, D_S^j
  Generate required abilities set $A_j^i \forall j \in S$
  Generate $C_S^i = \sum_{j \in S} c_j^i$.
  Determine $d_j^i = (U_T - C_S^i) \cdot \frac{1 - \frac{c_j^i}{C_S^i}}{|S|-1} \; \forall j \in S \Rightarrow D_S^i$.
  Evaluate d_i^j and d_i^i: $y = |1 - \frac{d_i^i}{d_i^j}|$.
  if $y \leq \alpha$ **then**
    respond with positive
  else
    respond with negative
  end if
end if

The proposer procedure is three-staged: the first stage calculates the anticipated benefit of each coalition for this agent. If the coalition is better than the coalition of just itself (d_i^i *singleton*) then in the second stage, all of the agents in the potential coalition are proposed the coalition with associated profit vector. Once all mutually acceptable coalitions are produced, in the third stage, the best coalition generated is sent to the other agents for final approval. If approved, the agents begin execution on the task with the formed coalition.

The responder handles messages from the proposer. It evaluates the message type, then proceeds according to the type. If the message type is for final approval, the responder evaluates the request in terms of any currently approved coalitions. If it is a proposal, it generates its own expected profit vector P_S^i then compares it to the offered payoff to determine whether the proposal is acceptable. The parameter α is user defined, and dictates the degree to which the models must agree before accepting a coalition. A reasonable value is 0.3, which requires that a coalition proposal must contain a profit for agent i which reflects i's expected profit within 30%. Smaller values for α require greater agreement in the models, which exponentially increases the number of times agents must interact before agreeing to proposals issued by other agents, all other things being equal.

The procedures presented above do not appear to address total quality directly. The quality is addressed during the generation of the abilities in the proposer's procedure. The value of each ability affects the quality of each agent. This affects the profit of the coalition, so the quality is captured by the procedure. The procedure favors smaller coalitions since the payoff is static for the task and the profit is divided among the performing coalition. This may often lead to singleton coalitions unless the ability sets are designed appropriately. A behavior may have multiple estimated costs associated with it, for example, solo cost and group cost. The group cost may be lower, encouraging formation of non-singleton coalitions. For example, consider a task for a coalition to explore a space. The potential total cost for the task is some scaled value of the total area. A singleton coalition takes the entire cost upon itself,

so its profit is $d_i^i = U_e - C_S$ with C_S large and $D_S^i = d_i^i$. Increasing the coalition to two identical agents cuts the cost in half for the original agent. Assume full knowledge of abilities is available. Then without abilities that are affected by being in a coalition, agents 1 and 2 share the cost so $c_1 = c_2 = \frac{c_i^i}{2}$ and $C_S = 2c_1$, so $d_i^i = D_S^i \cdot \frac{1}{2c_i^i \cdot \frac{1}{c_i^i}} = D_S^i \cdot \frac{1}{2}$. However, if the abilities are affected by groups, agents 1 and 2 still share the costs but let $c_1' = \frac{c_1}{4}$ due to the layout of the space and associated agent costs. Then $C_S = c_1' + c_2' = 2c_1' = 2\frac{c_1}{4} = \frac{1}{2}c_1$. With $P_S = U_e - C_S$, $P_S = U_e - \frac{1}{2}c_1$ whereas in the singleton coalition, $D_S^i = U_e - c_1$. This nets a higher total profit for the coalition and the individual profit is divided evenly, giving $\frac{D_S^i}{2}$ to each or $d_j^i = \frac{U_e - \frac{1}{2}c_1}{2} = \frac{U_e}{2} - \frac{c_1}{4}$. If payoff is 20 and c_1 is 16, the payoff of the singleton coalition is 4 whereas the joint coalition is 10-4 = 6. The profit is increased for both agents by having groups formed and ability sets that incorporate a lower cost when employed in a group. A system designer may choose to build a more complex model where an agent's quality is affected by both the task type and the number of agents in a proposal. This requires an expansion of the utility value generation and evaluation, but leads to a much more sophisticated system which could favor coalitions up to a given size or of at least a given size. The broadcast utility values may then also require adjustment so they are delivered for a specific coalition size.

Lines 16 to 20 in the responder's procedure (Algorithm 2) indicates acceptance of a coalition if the profit is within $(\alpha \times 100)\%$ of the estimated profit. If the profit is more than $(\alpha \times 100)\%$ away from the expected, the proposed coalition is declined. Since this mechanism is based upon agreement between the models in each agent, each coalition need only be proposed once. If even one agent declines the issued proposal, the coalition cannot be formed since the declining agent has a model that generates profit vectors too dissimilar to the proposer. Additionally, if two agents simultaneously propose a coalition, the responder can override its proposer. The priority is based on the agent number: the proposer is set as the lowest agent number when a responder receives a duplicate proposal. Even if the proposed profit is high,

this approach enforces a designed-in skepticism, which by nature obeys the rule that if it sounds too good to be true, it probably is.

3.7 Example

To illustrate the procedure in a more direct sense, consider the following scenario: three agents, each with three abilities, are in a collective tasked to push a box to a destination with a payoff of 50. Agent 1 has the abilities PUSH, GOTO, and WANDER, of qualities 0.7, 0.7, 0.9, respectively. Agent 2 has the abilities PUSH, PRECISION_NAV, and SURVEY, of qualities 0.8, 0.8, 0.6, respectively. Agent 3 has the abilities FLIGHT, SURVEY, and DRAG at qualities 0.8, 0.4, 0.6. Agent 1 expects a task cost of 10 in a group and 40 alone, based upon its internal state. Agent 1 expects to use PUSH and GOTO. The proposer procedure for agent 1 is:

1. Determine quality: 0.7. Send quality to the other agents.
2. Receive qualities, update models (an example is in Appendix B). Assume the models end up generating accurate qualities after the update.
3. Generate all possible coalitions involving agent 1, without singletons: 1,2,1,3,1,2,3.
4. for each coalition, calculate the coalition costs: C(1,2) = 26.78, C(1,3) = 28.86, C(1,2,3) = 41.36.
5. generate profit vectors: 13.34,16.66, 15.26,14.74, 6.01,8.29,5.7.
6. propose coalition 1,3 to agent 3.
7. when agent 3 responds with negative, propose 1,2 to agent 2.
8. agent 2 responds positively. Save 1,2.
9. evaluate coalition 1,2,3. Solo profit is 10, coalition 1,2,3 gives 6.01. Skip proposing of 1,2,3.
10. present agent 2 with 1,2 for final approval. Assume positive response from agent 2.

11. announce 1,2 to agents 2 and 3.

The other two agents run the same procedure. At the end, they handle deadlocks by selecting the lowest total cost coalition of any that were announced. This example provided a step-by-step walkthrough of a potential scenario. Adjustment of any parameters: task payoff, cost expectations, or ability qualities might cause a widely different result since the parameters affect the values of the profit vectors. The proposer handles uncertainty in this example by making assumptions about the quality of an agent's model quality and incorporation of noisy state data to determine the agent cost and coalition cost.

3.8 Notes

The computational complexity is $\mathcal{O}(n * 2^n)$ per agent(see Appendix A.3). The quantity of communication messages sent per agent is also $\mathcal{O}(n * 2^n)$ per agent (see Appendix A.2). This is not as time consuming nor as processing intensive as those procedures which generate the entire coalition structure. An example is Shehory and Krauss [35], which takes $\mathcal{O}(n^n)$. An in-depth evaluation of the communication message quantity and processing time are shown in Appendix A.

The algorithm assumes there are no external shared resources. This framework is designed for task allocation to a subset of agents, and the resources that are allocated are found entirely within the agents. All shared resources are intra-agent, and are limited to the environmental data and working variables that an agent's proposer and responder must access. This data, with the exception of the explicitly transmitted environmental data, proposals, responses, utilities, approval solicitation and responses, and final announcements, is completely internal to a given agent. Appendix A also contains a discussion about risks regarding the sharing of this data and how they are mitigated by this approach.

The tasks are processed individually and sequentially. Each agent may contain a set of tasks to process, but since it is assumed that every agent knows of each task,

the task sets are also assumed equal. Once an allocation of a specific task is made, the execution may begin immediately or be delayed for a time. This allows for another representation of the abilities sets regarding inactive or busy. Inactive agents may be more willing to join a coalition for a task than those agents already allocated to another task. This mechanism is applicable to ensure that every agent is occupied over a set of tasks, depending upon the severity of the adjustment between inactive and busy values.

The distributed nature of this approach is built to ensure robustness over the coalition formation process. A fully centralized or even temporarily centralized coalition formation mechanism built upon the protocol presented here would build a coalition upon the models within only one agent. This would help to create and enforce optimality based upon the one agent's expectations. However, if the allocating agent's model generates widely different results than the models in other agents, the coalition it defines as optimal is truly suboptimal. The fully distributed approach and the model convergence aspects of the algorithm ensure that, even if a formed coalition is suboptimal, no one agent is dominating the decision and forcing a suboptimal coalition onto other agents. This improves the likelihood that the formed coalition is optimal or at least core-stable.

3.9 Stability

The Bayesian core is applied as the stability model for coalition formation. This model states that a coalition structure is in the Bayesian core if each agent in the coalition structure does not expect that it can do better by changing coalitions and without harming another agent. Initially, each agent is in a singleton coalition: a coalition of only itself. Each agent i generates the estimated agent costs for a task. The agents use this information and information about the task to generate every coalition involving itself. It then computes, for each of the generated coalitions, the coalition cost and profit distributions. The profit distributed is always a positive value. It is this profit that each agent seeks to maximize. The agent must also

consider, however, that it cannot simply force a particular coalition: all agents in the proposed coalition must agree. That is, each agent in the proposed coalition must expect a greater profit than it expects to receive in another formable coalition. Given this profit distribution, the agent sends a proposition to the other members of a proposed coalition which includes the coalition members and the profit vector. The other members respond with either an affirmative or negative. If a negative response is sent, it is an egocentric response that indicates unsatisfactory distribution to the responder. This reflects a poor estimate of the responder's abilities as they are understood by the proposer.

The Bayesian core defined by Chalkiadakis and Boutilier [11] takes a cooperative view of the formation mechanism in that the agents only form coalitions if the resulting coalition structure is expected not to harm any agent. The definition of the core in game theory is less restrictive and is not concerned with harming other agents. The game theory representation requires that each agent feels it is receiving the best profit it can achieve. However, an agent cannot force a coalition to form, so the core implies that all agents in a proposed coalition (not the entire structure) are better off or at least no worse than in any other coalition. If this is not the case, at least one agent in a proposed coalition would disagree with the proposal, thereby not forming it. The assumption used here is egocentric in this manner: the other coalitions don't matter as long as the agent under consideration is receiving its best payoff, given the coalition structure.

Agents do not always form the optimal coalition. If a "very good" agent can receive a greater profit by working with a poorer quality agent, it prefers the poorer quality agent for a partner. However, as the profit margin diminishes, agents lean more towards the optimal coalition, since some profit is preferable to no profit. In practice, the core-stable solution is always chosen (unless poor models or communication failures prevent it from forming), since the core-stable coalition is the optimal coalition under very low profit margins.

3.10 Summary

Extending full uncertainty to coalition formation processes involves each agent performing processing on its internal state, rather than on the true state. If the agents have perfect knowledge of the state (which includes agent abilities, the environment, and the coalition structure), it becomes a process that can guarantee optimality (given the core stability concept). Even without full state knowledge, the procedure presented in Section 3.6 forms best coalitions based upon the internal state. The design of the ability representations has significant effect on the coalitions that are formed and the quality of the coalitions for a task. Repeated interaction of agents in coalitions and the formation processes improve the models of the agents through employment of a modified Kalman filter. The ensuing coalitions are robust: once a best coalition is formed, no better coalition can be found for a task. A member of a coalition can multitask by joining multiple coalitions through the sequential processing of tasks, since a coalition only addresses a single task and only a single coalition is assigned to a task. Upon completion (or failure) of the task, the coalition is disbanded. Variations on this and other features of the coalition formation procedure presented here are provided in the next chapter, which discusses implementation and experimental setup to test the mechanism.

IV. Software Agent Implementation

The formulation presented previously is applied as a solver in multiagent teams. Results from a specially developed simulation show the scalability of the mechanism and its handling of uncertainty. This simulation is used as a source for analysis and evaluation of the mechanism related to model convergence rates and the effects of agent quantity on the runtime and number of messages sent.

4.1 Description

A simulation was developed with a number of heterogeneous agents that must move an item. It is assumed that there are no subtasks. The ability types are FLIGHT, OBSTACLE_AVOID, GOTO_XY, PUSH, LIFT, MOVE_ITEM, ALL_STOP, PRECISION_NAV, VISION, RGB_VISION, GRAY_VISION, and RANGE. Each agent has a subset of these types, with both solo and group qualities, and the group execution quality is sent at the beginning of the CFDP. The agents are broken into three types. The quality is determined from the values for each ability type within an agent, i.e. two agents of identical type have identical quality for group execution in this simulation. Each agent type possesses at least one ability that the other agents do not. The full configuration of each agent type is shown in Tables 4.1, 4.2, and 4.3 for agents of type 1, 2, and 3, respectively. Defined through the tuple from Chapter III, the CFDP is:

- N: the set of agents in the collective. $N = 1, 2, ..., n, n \in \{6, 7, 8, 9, 10, 11, 12\}$.
- T: the task MOVE_ITEM.
- M_i: the set of abilities and values agent i uses to represent the other agents.
- Q_t^i: 0.91667, 0.88333, and 0.7833 for agents of type 1, 2, and 3, respectively.
- C_t^i: constant for this testing. Set to 1.0.
- $V_t^i(S, t, M_i, C_t^i)$: Solved for by the proposer and responder. Used as the measure of a coalition's quality.

Table 4.1: The configuration for agents of type 1.

Agent Type	Solo/Group	Ability Type	Quality
1	Solo	ALL_STOP	0.6
		GOTO_XY	0.2
		LIFT	0.8
		MOVE_ITEM	1.0
		OBSTACLE_AVOID	0.2
		RANGE	0.3
		RGB_VISION	0.04
	Group	ALL_STOP	1.0
		GOTO_XY	0.8
		LIFT	1.0
		MOVE_ITEM	1.0
		OBSTACLE_AVOID	0.9
		RANGE	0.8
		RGB_VISION	1.0

The set of tests performed here investigates the effects of M_i on the coalition formation.

Agents of type 2 and 3 tend to propose coalitions, whereas agents of type 1 tend to decline coalitions since their profit is not increased by joining a group. The effects of different quantities of each agent type and the quality of each agent's internal model of the other agents are investigated. The factors of interest are the computation time and the number of communication messages sent. The simulation is designed such that communication does occasionally fail and multiple agents may issue proposals to each other at the same time.

The ability types that are used to execute the move item task include OBSTACLE_AVOID, GOTO_XY, PUSH, ALL_STOP, RANGE, LIFT, FLIGHT, and MOVE_ITEM. Each agent contains a subset of these abilities, so their contributions are based upon their subset.

The testing is broken into two parts: perfect models and imperfect models. The simulation has no true environment, so each agent assumes a constant cost of 1.0 for tasks. The effects of varying the quantity and type of agents on the computation

Table 4.2: The configuration for agents of type 2.

Agent Type	Solo/Group	Ability Type	Quality
2	Solo	ALL_STOP	0.1
		GOTO_XY	0.1
		PUSH	0.1
		MOVE_ITEM	0.1
		OBSTACLE_AVOID	0.1
		RANGE	0.1
		RGB_VISION	0.1
		PRECISION_NAV	0.1
	Group	ALL_STOP	1
		GOTO_XY	0.8
		PUSH	1.0
		MOVE_ITEM	0.8
		OBSTACLE_AVOID	0.9
		RANGE	0.8
		RGB_VISION	1.0
		PRECISION_NAV	1

Table 4.3: The configuration for agents of type 3.

Agent Type	Solo/Group	Ability Type	Quality
3	Solo	ALL_STOP	0.04
		MOVE_ITEM	0.04
		GOTO_XY	0.08
		FLIGHT	0.1
		OBSTACLE_AVOID	0.09
		RANGE	0.06
		GRAY_VISION	0.07
		PRECISION_NAV	0.07
	Group	ALL_STOP	0.8
		MOVE_ITEM	0.6
		GOTO_XY	0.6
		FLIGHT	0.9
		OBSTACLE_AVOID	0.9
		RANGE	0.9
		GRAY_VISION	0.9
		PRECISION_NAV	1.0

time and communication message quantity are tracked in the perfect model testing. The imperfect model testing evaluates convergence rates: the quantity of repeated evaluations of a task needed for the models to converge sufficiently to form coalitions.

4.1.1 Perfect Model Testing. With perfect models, the agents can evaluate and propose potential coalitions to any other agent, confident that the other agents will accept if the other agents cannot receive a better payoff alone. The ability types and values are perfectly captured, so agents of any type know that agent 1 is of type 1 and uses the abilities OBSTACLE_AVOID, GOTO_XY, ALL_STOP, RANGE, LIFT, and MOVE_ITEM for the task. The other agents also know the quality of each of these abilities for group execution. They do not, however, know how these qualities compare to the solo execution qualities.

The effects of a specific type of agent on the message quantity and processing time may be significant, depending upon the types of the other agents in the collective. For example, a 12 member collective composed solely of agents of type 1 on a standard task completes rapidly since no agents issue proposals. However, changing this setup to a 12 member collective containing 4 agents of type 1 takes an average of 4,821 seconds to complete. The configurations selected for testing contain a split between the agent types in order to examine the worst case scenario for processing time. The quantity of each agent type for each run is shown in Table 4.4. Each agent has two values for each ability: solo and group. The solo value gives the quality if an agent chooses to work alone, and the group value gives the quality if the agent chooses to work in a group. This approach favors two-member coalitions, though it does not prevent three or four member coalitions in the case of poor models or communication failures.

Fig. 4.1 indicates the time taken for each run. The dots are realized run times and the line indicates the trend. Note that the plot is semi-logarithmic, reflecting the 2^n effects on the growth rate when an agent is added. The runtime effects are not

Table 4.4: The run numbers with the quantity of each agent type. Each run was executed 10 times.

Run	Type 1	Type 2	Type 3	Total
1	2	2	2	6
2	3	2	2	7
3	2	3	2	7
4	2	2	2	7
5	4	2	2	8
6	2	4	2	8
7	2	2	4	8
8	5	2	2	9
9	2	5	2	9
10	2	2	5	9
11	4	4	4	12

significantly affected by the agent types, though this is due to the mixture of types in the collective.

4.1.2 Imperfect Model Testing. This test evaluates the number of times a task has to be re-evaluated before the agents agree to a coalition. The convergence rate when the models are imperfect depends greatly upon the initial quality of an agent's models relative to the true configuration of the other agents. If the models are of poor quality (as in, the result of inferring a quality from the ability values is vastly different from the agent's true quality), the agents may have to process a task multiple times prior to being able to form a coalition, since their models improve when the utilities are sent each time a task is presented. Each agent's unique abilities are not represented by the agents of any other type. In other words, an agent of type 1 assumes that all other agents use OBSTACLE_AVOID, GOTO_XY, ALL_STOP, RANGE, LIFT, and MOVE_ITEM for the task, even if the other agents do not possess those skills[1]. The agents are set to initially assume a quality of the other agents' abilities as shown in Table 4.5. Those values are assumed the same for all

[1]Even if the agent represents another agent with an accurate set of behaviors but with poor quality estimates, the nature of the experiment is the same. The convergence times are no different in either case.

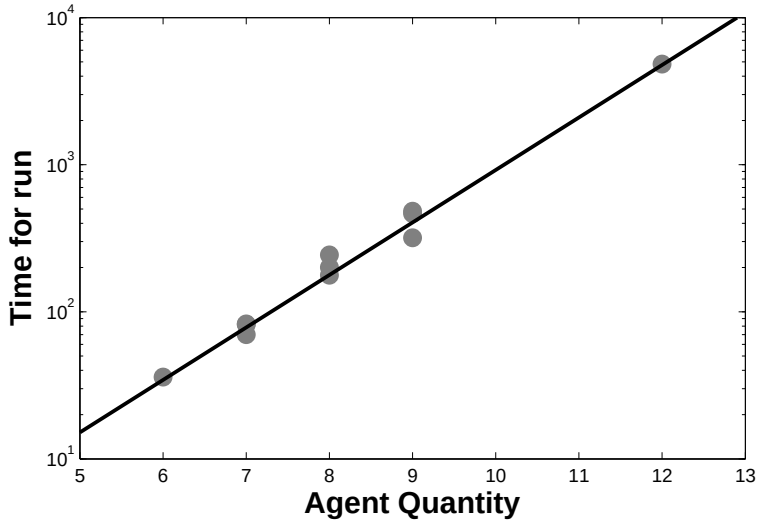

Figure 4.1: Semi-logarithmic plot showing the effect on runtime of adding an agent. The mean values for each quantity of agents is shown relative to a linear regression fit. Each agent has perfect models of all other agents.

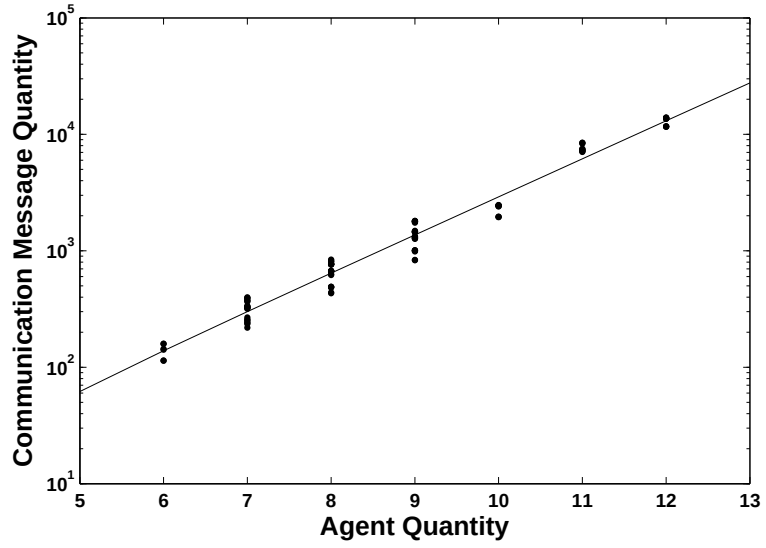

Figure 4.2: Semi-logarithmic plot showing the effect on message quantity of adding agents. A linear regression fit is also shown. The growth rate is governed by 2^n, though these values are about four times higher than the runtime.

Table 4.5: The assumed initial quality of the other agents for the imperfect model testing. For each run, each agent makes the assumption that all other agents have the corresponding initial quality value for all abilities.

Run Number	Assumed Ability Qualities
1	0.01
2	0.05
3	0.1
4	0.2
5	0.3
6	0.4
7	0.5
8	0.6
9	0.7
10	0.8
11	0.9
12	1.0

abilities (as in, each $a_\tau^{ij} = 0.01 \; \forall \tau \; \forall j$ in the first run). The ability variances are initially set to 0.5 and the measurement noise variance is set to 0.2. A six-agent collective with two of each agent type is used. For each run, a task is re-evaluated upwards of 10 times. In real-world scenarios, another opportunity for learning would be through performing an evaluation of the task after execution is complete, which would cause the convergence rate to become partially dependent upon the formed coalition. The models were tested on a number of different configurations. The qualities that the agent types provide for this task are 0.91667, 0.88333, and 0.7833 for types 1, 2, and 3, respectively. These qualities are used to generate residuals, which in turn reflects a model accuracy. For example, run number 11 in Table 4.5 has residuals of 0.01667, 0.01667, and 0.11667 whereas run number 1 has residuals of 0.90667, 0.87333, and 0.7733 for the first exposure to the task. Since the residuals are the only external source of insight into the model quality, run 11 has more accurate models than run number 1.

Since agents only agree to form coalitions if their internal model agrees (within 30%) with the issued proposal, the number of task re-evaluations it takes before agree-

ing on a coalition may be long. Fig. 4.3 shows the number of re-evaluations of a task required to generate coalitions. Once the models generate qualities within 0.1 of the true qualities, a coalition is formed. The 30% threshold parameter ($\alpha = 0.3$) defines this requirement. The initial models that generate more accurate results converge faster (runs 9-12 formed coalitions in the first evaluation), but even very poor initial models generate acceptable coalitions after a few iterations. More complex domains with widely varying task types may cause the convergence to take more iterations, but convergence occurs regardless of initial model quality. Other parameters, such as the noise variance and the ability variances affect convergence rates, yet also affect the degree of confidence. In other words, increasing the initial variance for the abilities provides faster convergence in the state vector, yet slower convergence in the covariance matrix. Increasing the value for the measurement noise also provides faster convergence in the state vector yet establishes a higher "lower limit" to the covariance. These effects are investigated in Appendix B.

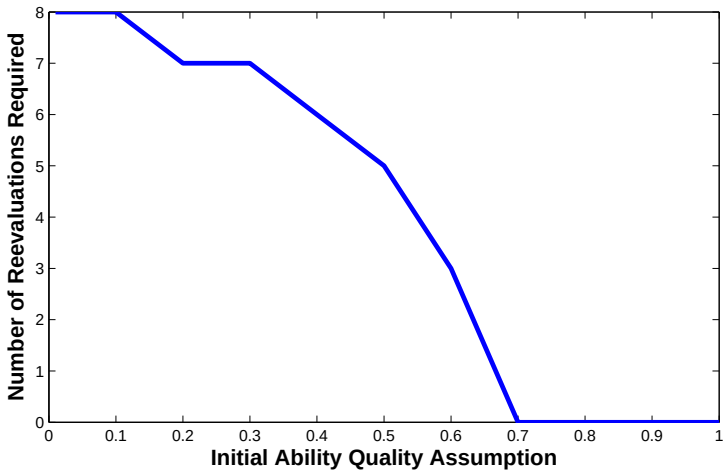

Figure 4.3: The convergence rates of the models seeded at different initial ability qualities.

4.2 Summary

This chapter presents information on the scalability and convergence of the approach. The runtime scales proportional to the number of agents performing the

negotiation. The actual runtime is highly dependent on the processor in the computer system: the testing for this chapter was performed on a single core mobile processor running at 1.58 GHz. Later testing was performed on a quad-core processor running at 2.1 GHz. The runtime in the latter case was approximately $\frac{1}{6}$ of the runtime from the former. The convergence results demonstrate the customizability of the approach on a per-agent basis. A system designer could create an agent whose initial models are very poor, and the nature of the agent model update technique establishes a means to gradually improve these poor models. In practice, it is rare to require renegotiation on a task more than four times before an agent is willing to join most of the proposed coalitions. The analysis provided in this chapter defines runtime and convergence, but does not comment on the optimality of the formed coalitions. Optimality is addressed in Chapter V. Chapter VI describes the implementation of the stochastic model and algorithm on real-world robotic systems.

V. Comparison

The biggest strength of the combination of the stochastic model and coalition formation algorithm is the ability to form coalitions under uncertainty. The uncertainty in agent type that the presented approach can handle is significant, permitting incorporation and assimilation of previously unknown agent types without reprogramming any of the original agents in the collective. To evaluate the uncertainty handling in a quantitative manner, the approach is tested against the approach from Chalkiadakis and Boutilier [11], which is also designed to handle uncertainty.

5.1 Setup

The algorithm as presented in Chapter III was applied to a simulated environment as defined by [11] and tested against their approach. The environment consists of 5 agents, each having a specific type: 0, 1, 2, 3, and 4 with quality values of -1, 1, 2, 3, and 4, respectively. The re-scaled variant of this to fit the framework of the algorithm from this paper defines the quality values at 0.5, 0.6, 0.7, 0.8, and 0.9, respectively. Communications may fail randomly, and failed communications may prevent a coalition from forming. The levels of uncertainty in agent type are changed in each test, and the total number of times per run the optimal or core-stable coalition is formed is tracked. The optimal coalition is defined as the most profitable coalition overall, i.e., the coalition involving only the two best agents. The core-stable coalition is the most profitable for the best agent in the collective. These agents form a coalition to execute a medium-sized project, where the payoff is set at 40. The test is run 30 times, with 100 iterations each time. The evaluation is broken into three overall tests:

1. No uncertainty in agent type, agent cost, or coalition cost.
2. Uncertainty in agent type, but all agent types come from a known set.
3. Uncertainty in agent type, and at least one agent type falls outside a known set.

Two additional agents are added for test three, agents 5 and 6. In tests one and two, the optimal coalition is with agents 4 and 3, and the core-stable coalition contains agents 4 and 1. For test three, the optimal coalition is with agents 6 and 4, with the core-stable containing agents 6 and 1.

5.2 Results

For the first test, Chalkiadakis' approach converged to either the optimal coalition or the core-stable coalition 22/30 times (due to communication failure and the constraints introduced by the random selection of the single proposer). The mean number of times per run that Chalkiadakis' approach formed either of these coalitions is 64.1, with a standard deviation of 37.5. The algorithm presented here formed it a mean of 83.0 times, with a standard deviation of 5.3, where each failure is due to communication failure. The implications of these results are the improved consistency of the algorithm presented here over Chalkiadakis' approach. Figure 5.1 shows a comparison of the two. The approach presented by Chalkiadakis either converged successfully or failed to entirely. There is no moderation in the results, and the polarity of the plot showing the results of applying Chalkiadakis' approach indicates this. The algorithm presented here has more consistent results, indicated by the much smaller standard deviation.

In the second test, uncertainty in agent type is introduced. The agents have initially uniform knowledge of the types of other agents, and the types are all taken from a set of known possibilities for Chalkiadakis' algorithm. Chalkiadakis' algorithm converged to the optimal or core-stable coalition a mean of 57.8 times with a standard deviation of 32.9. The algorithm presented here has initial assumptions of 0.5 for all behavior quality values, 0.5 for standard deviations on the behavior qualities, 0.3 for the threshold (α) values, and 0.3 for the measurement noise (v). This algorithm formed either the optimal or core-stable coalition a mean of 84.4 times, with a standard deviation of 4.3. These results compare favorably with the results of the first test. The advantage comes with the short term required to make the models agree. The

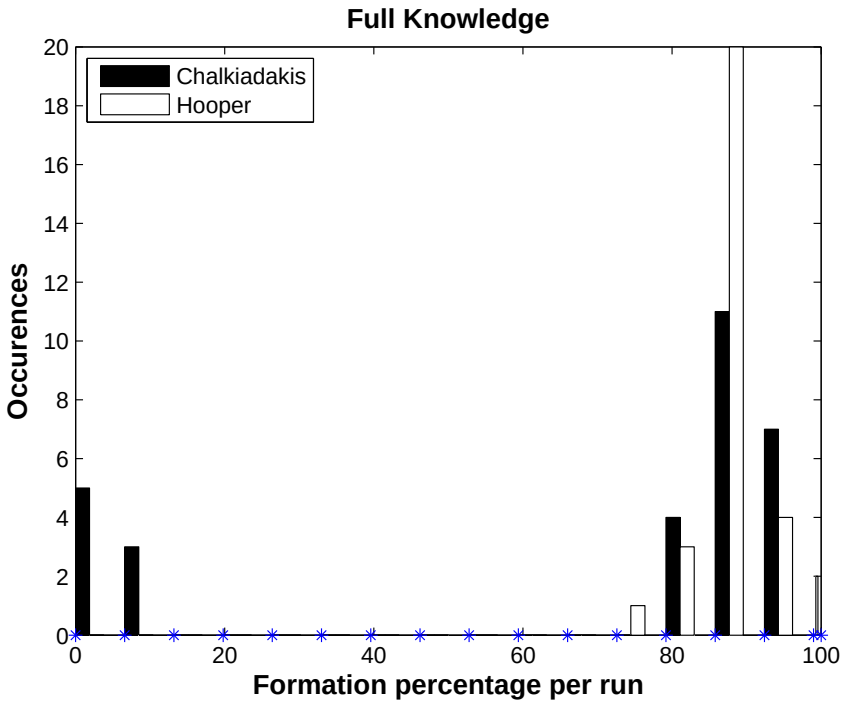

Figure 5.1: A comparison of the two algorithms under full knowledge of all agent types.

model agreement is finalized in no more than five iterations, so agents are forming coalitions and improving the quality of these coalitions for only a short time. After the first five iterations per run, the algorithm continues as if it is in the same situation as the full knowledge test. The random communication failures were more significant in the first test, so the results for both test one and test two are very similar. Figure 5.2 shows the difference in the uniformity of the results from each algorithm.

In test three, uncertainty is increased further. Two agents are added (agents 5 and 6), each with a type that falls outside the known set. Each new agent can represent itself and the previous set accurately, but not the other new agent. The pre-existing five agents can only represent the known set (the other preexisting agents) accurately. The new agents have quality values of -2 and 6 for the representation in Chalkiadakis' algorithm, and 0.4 and 1.0 for the representation in the algorithm from this paper. Chalkiadakis' algorithm converged to either the optimal or the core-stable coalition a mean of 36.8 times with a standard deviation of 34.9. The algorithm presented here

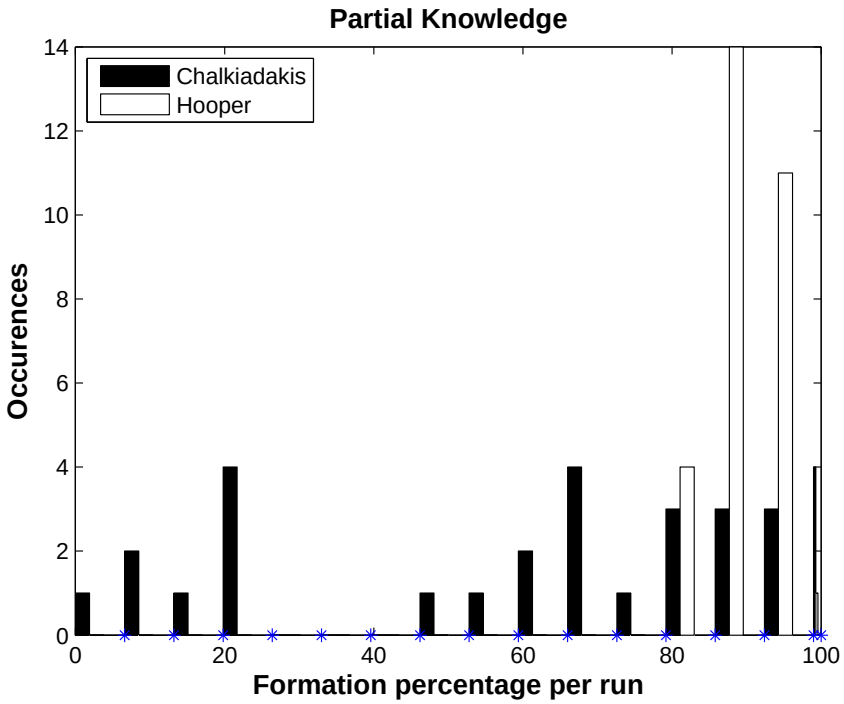

Figure 5.2: A comparison of the two algorithms with uncertainty in agent types.

formed it a mean of 82.8 times with a standard deviation of 8.4. This test shows the strength of the algorithm. The new agent types have not previously been encountered by the other agents, yet the new agents are fully assimilated in five or fewer iterations. A comparison of the results of the algorithms is presented in Figure 5.3.

5.3 Summary

The results of tests 1, 2, and 3 indicate that the algorithm and stochastic model presented here not only allows for uncertainty on a larger scale, but also performs more consistently even in the face of a small amount of uncertainty. The open-ended representation allows agents of arbitrary types to be added to an agent collective and, provided all agents are running the same framework, provides rapid assimilation and minimizes downtime. This is uniquely suited to collectives with high turnover, i.e., new agent types are introduced fairly often. The ability to handle these arbitrary agent types and yet perform well when all details about the other agents are known

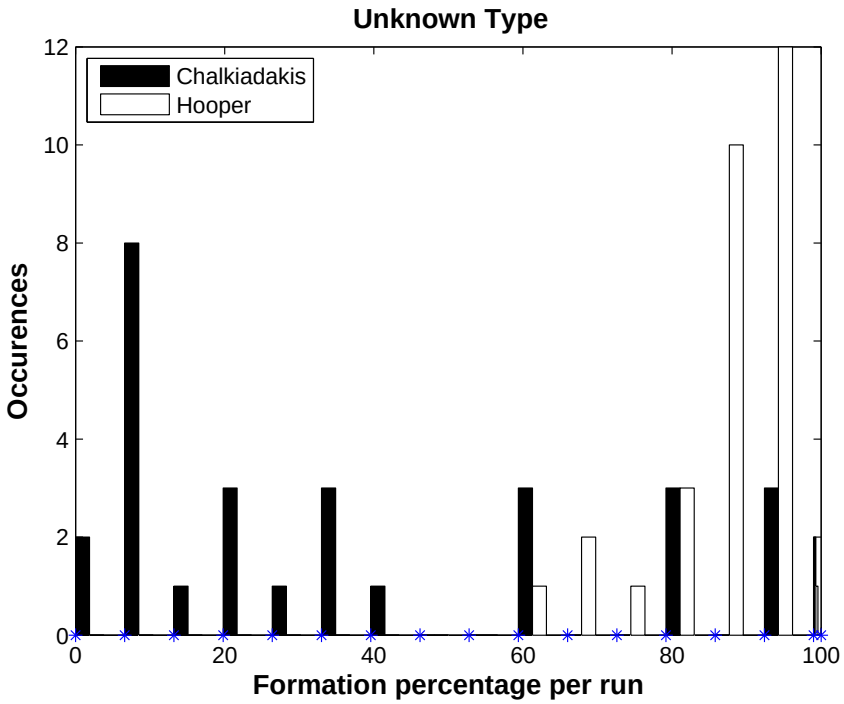

Figure 5.3: A comparison of the two algorithms with two unknown agent types.

improves the strength and flexibility of task allocation. Even in the face of failed communication, unknown agent types, and distinctly poor initial models, the approach operates well and improves over time. This chapter indicates the effects of moving from full knowledge of other agent types through uncertain agent types to completely unknown agent types. The operation is consistent, having a normal distribution and managing at least 80% formation rate regardless of the level of uncertainty. This indicates the strength of the approach despite significant agent type uncertainty.

VI. Robotic Implementation

The application of the model and algorithm to real-world robot systems demonstrates its flexibility. Since robotic collectives may be composed of a wide variety of robots (e.g., humanoid and other bipedal robots, skid-steer style robots, and quadripedal robots), the ability to generalize an approach to handle random robot types is important. The model and algorithm application to a real world robotic collective help expand the potential ubiquity of coalition formation applied to robotic collectives, since it is more generally applicable under the representation in this dissertation. This chapter provides a description of the deployment, including testing that demonstrates the ability to add an arbitrary robot type to a preexisting collective. The algorithm is applied to four robots: three Pioneer P2-ATs and one Lego Mindstorms NXT, shown in Figure 6.1 executing the MOVE_ITEM task. The Pioneer robots use the HAMR control architecture [26], which is built on top of Player/Stage [22]. The NXT is programmed using LeJOS with the Unified Behavior Framework [45] and a simple, static planner. The robots allocate three tasks which would be typical of a search-and-retrieve activity. These tasks include the exploration of an area, the locating of an item, and the movement of the item to a goal position. These tasks are broken into three task types in the deployment: MOVE_ITEM, FIND_ITEM, and EXPLORE. For MOVE_ITEM, robots push the item to the goal position. Better execution of this task is dependent upon a robot's torque and the nature of its front (i.e., whether a bumper or gripper is present). This task is tightly coupled in its execution. The FIND_ITEM task requires robots to locate a red cylinder in the environment. This task is dependent upon robot speed and sensor ability, requiring a color camera. The EXPLORE task requires robots to explore the environment. This requires velocity control and obstacle detection. The FIND_ITEM and EXPLORE tasks are loosely coupled in their execution. Including both tightly and loosely coupled tasks demonstrates the flexibility of the allocation method.

Figure 6.1: The robotic collective executing the MOVE_ITEM task. The two robots in the foreground are Frodo and Gandalf, the robots in the background are Gollum and the NXT.

6.1 Robots

The NXT is a skid-steer treaded robot with two Mindstorms sonar sensors. It cannot detect color, nor does it have enough torque to push an item. The NXT has five behaviors: obstacle avoid, wander, push, view, and GoToXY. Each behavior is an ability in the coalition formation algorithm. A combination of these abilities are selected depending on the task the robots are negotiating on, and it assumes that the other robots each have the same five abilities. This set of five abilities is approximately 20% of the size of the ability sets on the three Pioneer 2ATs: Gandalf, Frodo, and Gollum. Gandalf has a color camera with blobfinder, a 180° LiDAR, 16 sonar sensors, a flat-front bumper, and 25 behaviors. Frodo has a vision-aided navigation system, 180° LiDAR, 16 sonar sensors, a flat-front bumper, and 36 behaviors. Gollum has a color camera with blobfinder, 180° LiDAR (not used), 16 sonar sensors, a 2-DOF gripper, and 43 behaviors. This varying hardware affects the quality of each robot for the tasks as described above. The quality values for these robots for each task type are shown in Table 6.1. Each of the robots represent the others in their own

Table 6.1: Robot Quality Values and Thresholds

Task Type	Gandalf	Frodo	Gollum	NXT
MOVE_ITEM	0.9	0.692	0.6	0.27
EXPLORE	0.72	0.73	0.6	0.7
FIND_ITEM	0.9	0.578	0.9	0.275
Threshold (α)	0.4	0.2	0.3	0.5

context: Frodo assumes all of the other robots have vision-aided navigation systems, 180° LiDARs, 16 sonar sensors, and flat-front bumper like itself. All of the robots use their own version of this representation, and there are abilities associated with their representation that other robots do not possess. For example, Gollum has the ability to individually move a small and light item, but it has no ability to use its laser to navigate around obstacles. HAMR receives the allocation and manages the execution of tasks for the Pioneer robots, and the static planner manages the execution of tasks for the NXT.

6.2 Experimental setup

Initially, only Frodo, Gandalf, and NXT are available for tasking and negotiate five times each on the three tasks. Iterations 1-5 are on the MOVE_ITEM task, where each robot evaluates the task and sends the appropriate quality value. After the fifth iteration, the task changes to the EXPLORE task (iteration 5-10), and the quality values sent out are based on the new task. The last five iterations are on the FIND_ITEM task. The environment the robots act in is a hallway two meters wide. This hallway follows a rectangular path of 50 by 30 meters. All robots have the same position origin in two-dimensional space. The environment is not static: the internal environment maps are removed from each robot before each task negotiation, and there is no control over the location, volume, or movement speed of pedestrians in the environment. To show the ability to randomly add a robot to the collective, Gollum is introduced after this 15-negotiation indoctrination period and the four robots proceed

with nine more negotiations. Iterations 1-3 are for the MOVE_ITEM task, 4-6 are on the EXPLORE task, and 7-9 are on the FIND_ITEM task. The robots execute the task after each iteration if a coalition is formed.

The *optimal* coalitions (the coalitions with lowest total cost) for each task are with Gandalf and Frodo for the MOVE_ITEM task, Gandalf and Gollum for the FIND_ITEM task, and Gandalf and Frodo for the EXPLORE task. The *core-stable* coalitions (the coalitions with the most profit available for its members) are with Gandalf and Gollum for the MOVE_ITEM task, Gandolf and Gollum for the FIND_ITEM task (since both the NXT and Frodo are very poor at the task), and Frodo and Gollum for the EXPLORE task. The robots form the core-stable coalition only if the profit margin is sufficient to justify it. The optimal coalition has a lower overall cost, but due to the nature of the core stability concept, the robots prefer increased profit.

6.3 Results

The threshold values (α) defined in Table 6.1 are each robot's tolerance for coalitions that don't match their internal stochastic models. Since the NXT's threshold value is 0.5, it has a high tolerance and is generally optimistic, i.e., it has a greater faith in the quality of the models that the other robots are working from. Frodo, on the other hand, is pessimistic, holding a higher trust in its own models. These differences shape the trend for the first few negotiations on the tasks, with Frodo declining every proposal (that it does not issue) for the first two negotiations and the NXT accepting every proposal for all of the negotiations. The robots tend to prefer to form the core-stable coalition for execution of a given task. Table 6.2 indicates the core-stable and optimal coalitions for each task type.

The changes in the residuals over the first 15 negotiations are shown in Figures 6.2, 6.3, and 6.4 for Gandalf, Frodo, and the NXT robot, respectively. The plots reflect the robot's expected quality and the actual communicated quality for each model. The first through fifth iterations are for the MOVE_ITEM. Due to the model's representation and operation, residuals diminish as the task is revisited. At the sixth

Table 6.2: Core-stable and optimal coalitions for each task type

Task Type	core-stable	optimal
Before addition of Gollum		
MOVE_ITEM	Gandalf Frodo	Gandalf Frodo
EXPLORE	Frodo NXT	Frodo Gandalf
FIND_ITEM	Gandalf NXT	Gandalf NXT
After addition of Gollum		
MOVE_ITEM	Gandalf Gollum	Gandalf Frodo
EXPLORE	Frodo Gollum	Frodo Gandalf
FIND_ITEM	Gandalf Gollum	Gandalf Gollum

iteration, the task changes, defining new residuals. These values change based on the task under consideration and the expected use of the abilities.

When Gollum is added to the collective, the other robots have well-established models of each other, so the residuals indicated by Figures 6.5, 6.6, and 6.7 change little by comparison, with the exception of the models for Gollum. The NXT robot is working from a small ability set, containing only 5 types of abilities. Frodo, Gandalf, and Gollum are working off ability sets between 22 and 30 elements in size, making their representation much more sophisticated. This expanded sophistication provides only moderate gain in terms of model accuracy, but allows each robot to maintain a representation over which it can reason independently of the other robots.

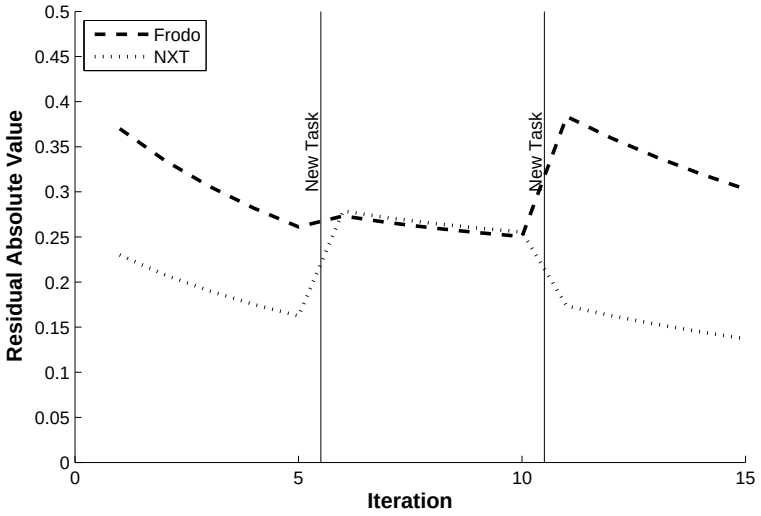

Figure 6.2: Residuals in Gandalf's models over the 15 negotiations prior to the addition of Gollum.

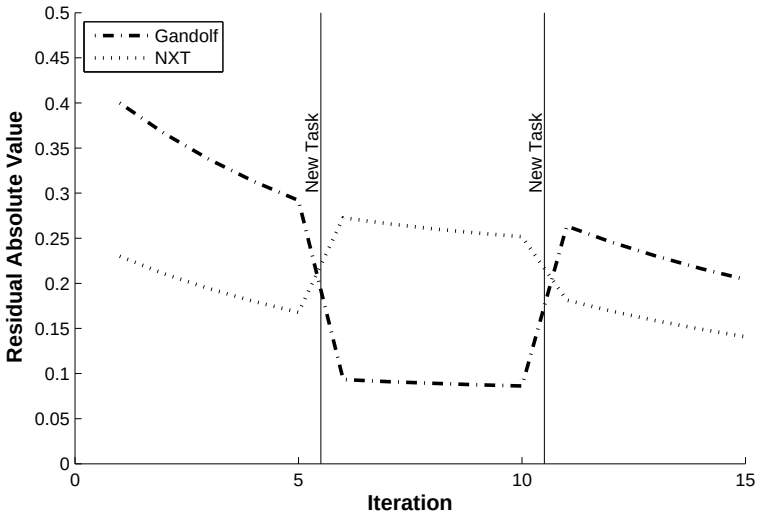

Figure 6.3: Residuals in Frodo's models over the 15 negotiations prior to the addition of Gollum.

Gollum's model residuals over nine negotiations after being added to the collective are shown in Figure 2.12. The residuals at the end of the nine negotiations reflect a similar end point as those of Frodo, Gandalf, and the NXT robot while having less time to fully indoctrinate.

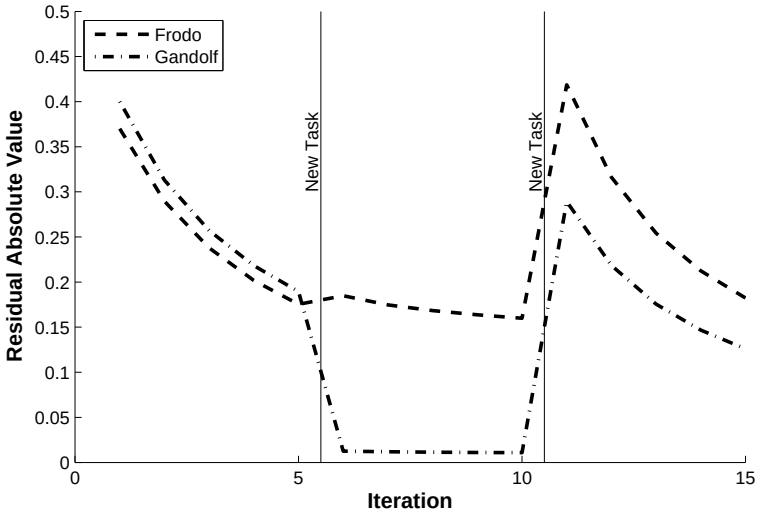

Figure 6.4: Residuals in the NXT robot's models over the 15 negotiations prior to the addition of Gollum.

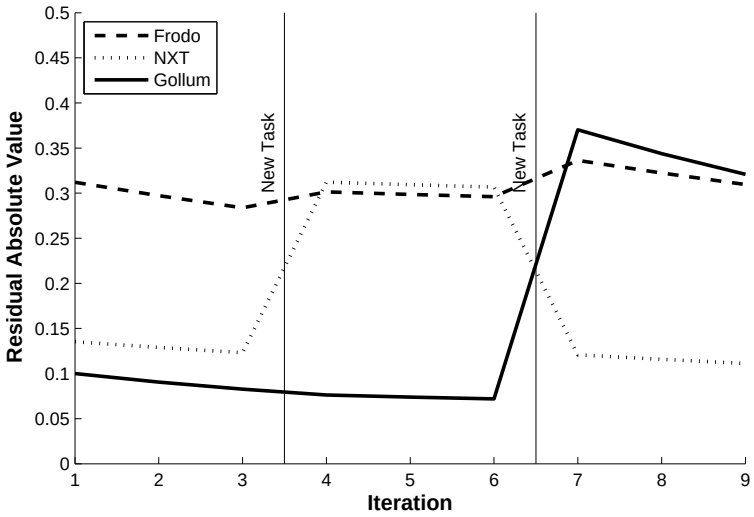

Figure 6.5: Gandalf's residuals over the 9 negotiations after the addition of Gollum.

In each iteration, every robot negotiated, proposing coalitions and expected payoffs to potential coalition members. Due to the threshold settings (shown in Table 6.1), some proposals were declined and others were accepted. The NXT robot accepted every coalition proposed to it unless it was offered a payoff of 0. Table 6.3 shows the coalitions formed with each iteration. When Gollum is added to the

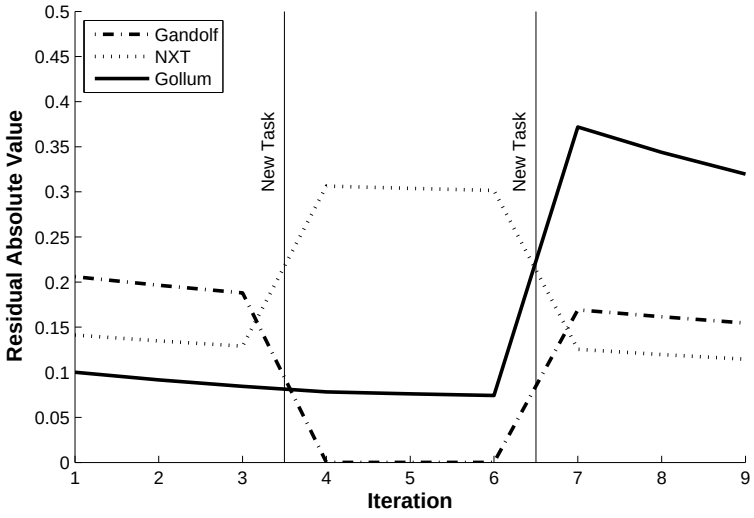

Figure 6.6: Frodo's residuals over the 9 negotiations after the addition of Gollum.

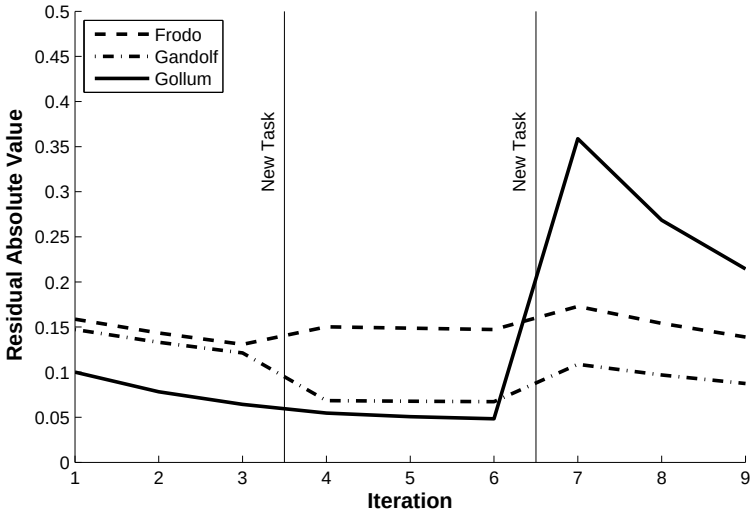

Figure 6.7: The NXT robot's residuals over the 9 negotiations after the addition of Gollum.

coalition, the core-stable coalitions for FIND_ITEM changes. In fact, the core-stable coalition and the optimal coalition are identical with the full 4-member collective for the FIND_ITEM task. This coalition is not formed until the third negotiation on the FIND_ITEM task, however, since the models of Gandalf and Gollum do not agree sufficiently for Gollum to accept Gandalf's proposals until then. Some iterations in

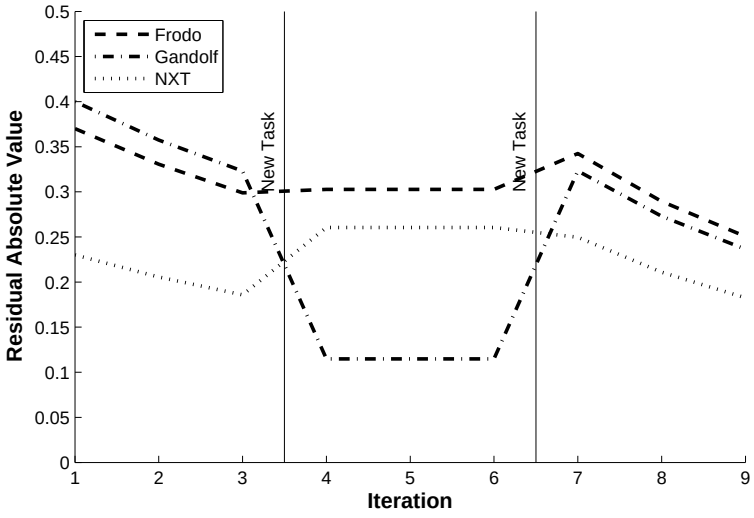

Figure 6.8: Gollum's residuals over 9 negotiations after added to the collective.

Table 6.3 show that no coalition was formed. This failure is due to communication failure or an underdeveloped model (e.g., Gandalf issued a proposal that was too far from Frodo's expectations), not uncommon in real-world robotic systems. The robots attempt to re-send a proposal up to 10 times to encourage a response from their targeted coalition members. Occasionally, when the models disagree to a certain extent and communications fail extensively, no coalition is formed. Iterations 3, 11 and 13 without Gollum and iterations 7 and 8 with Gollum fail to form coalitions. This is caused by two things: 1) the qualities of the robots that Gandalf has available to work with before Gollum is added are very poor, and 2) Gollum refuses Gandalf's proposals until the third iteration on the FIND_ITEM task (ninth iteration overall).

6.4 Summary

This chapter demonstrates the application of the model and algorithm to a robot collective. The strengths of the approach are indicated by the ability of the robots to form the core-stable coalitions despite the fully decentralized nature of the algorithm. It also demonstrates the ability to add a new robot type to a preexisting collective and adjust the task allocation appropriately. Before fully assimilated, Gollum does not

form coalitions. This is due to the disagreement of the models and the threshold (α) values. Once the models agree, Gollum agrees to proposals from the other robots (and, accordingly, the other robots agree to Gollum's proposals) and the new core-stable coalition is formed. The model allows this rapid assimilation and the flexibility to create and work with general collectives, regardless of hardware or past interactions. As a task allocation method, this approach is more robust and more flexible than many other approaches.

Table 6.3: Coalitions formed for each iteration

Iteration	Coalition Formed	Estimated Payoff	Actual Payoff
1	NONE	N/A	N/A
2	NXT/Gandalf	1.30, 31.83	5.48, 27.74
3	NONE	N/A	N/A
4	NXT/Gandalf	1.46, 31.64	5.48, 27.74
5	NXT/Gandalf	2.14, 30.82	5.48, 27.74
6	Gandalf/Frodo	19.48, 17.43	20.76, 15.84
7	Gandalf/Frodo	20.04, 16.78	20.76, 15.84
8	NONE	N/A	N/A
9	Gandalf/Frodo	21.03, 15.55	20.76, 15.84
10	Gandalf/Frodo	20.86, 15.69	20.76, 15.84
11	NONE	N/A	N/A
12	NXT/Gandalf	8.76, 27.33	7.23, 29.20
13	NONE	N/A	N/A
14	NXT/Gandalf	7.74, 28.10	7.23, 29.20
15	Gandalf/Frodo	22.83, 14.37	25.36, 12.65
After Adding Gollum			
1	Gandalf/Gollum	19.33, 17.0	23.39, 13.57
2	Gandalf/Gollum	20.49, 16.01	23.39, 13.57
3	Gandalf/Gollum	21.32, 15.29	23.39, 13.57
4	Gandalf/Gollum	19.51, 16.95	20.54, 16.12
5	Gandalf/Frodo	20.70, 15.90	20.76, 15.84
6	Gandalf/Frodo	20.76, 15.84	20.76, 15.84
7	NONE	N/A	N/A
8	NONE	N/A	N/A
9	Gandalf/Gollum	17.41, 20.18	18.91, 18.84

VII. Conclusion

This dissertation presents an improvement to previous approaches to coalition formation mechanisms with uncertainty. It defines a CFDP that is applicable to an unstructured environment and extends multi-robot task allocation and dynamic coalition formation to allow uncertainty and dynamic changes in the collective's composition. The simple yet robust stochastic modeling approach is applied to robotic and agent collectives to demonstrate similar runtime, expanded applicability, and better results than other approaches under uncertainty.

This dissertation presents the game theory perspective on coalition formation, discussing the three main stability concepts. It also discusses the distributed artificial intelligence approaches, some of which address uncertainty. Unfortunately, even with handling of uncertainty, the applicability of the coalition formation approaches to real-world robotic systems is limited. Thus, an investigation of POMDP's and multi-robot task allocation methods is provided. Furthermore, an in depth discussion of nontransferable utility is provided, which is a critical assumption made by many coalition formation approaches yet is invalid for most real-world applications. Under these constraints, a formalization of the Coalition Formation Decision Process (CFDP), the stochastic modeling technique, and the algorithm for forming coalitions are provided. The modeling approach enables application to dynamic coalitions where each agent performs processing on its belief state and needs not perfectly represent the other agents. Repeated interaction of agents in coalitions and the formation processes improve the beliefs of the agents through employment of a modified Kalman filter. The ensuing coalitions are robust, and a member of a coalition multitasks by joining multiple coalitions.

The model and algorithm are applied to an agent simulation and an investigation on the scalability and convergence of the approach are provided. The runtime scales proportional to the number of agents performing the negotiation, with an exponential growth rate. The convergence results demonstrate the customizability of the approach

on a per-agent basis. It is rare to require renegotiation on a task more than four times before an agent is willing to join most of the proposed coalitions.

The optimality and general solution quality of the approach are also presented. These results indicate that the approach allows uncertainty on a larger scale and performs more consistently even in the face of a small amount of uncertainty. The open-ended representation allows agents of arbitrary types to be added to an agent collective and, provided all agents are running the same framework, provides rapid assimilation and minimizes downtime. The ability to handle these arbitrary agent types and yet perform well when all details about the other agents are known improves the strength and flexibility of task allocation. Even in the face of failed communication, unknown agent types, and distinctly poor initial models, the approach operates well and improves over time. That chapter indicates the effects of moving from full knowledge of other agent types through uncertain agent types to completely unknown agent types. The operation is consistent, having a normal distribution and managing at least 80% formation rate regardless of the level of uncertainty. This indicates the strength of the approach despite significant agent type uncertainty. Further results demonstrate the application of the model and algorithm to a robot collective. It demonstrates the ability to add a new robot type to a preexisting collective and adjust the task allocation appropriately. The model allows this rapid assimilation and the flexibility to create and work with general collectives. The strength of the model is demonstrated via the real-world application and the late addition of Gollum to the collective.

In summary, this dissertation provides a combination of a stochastic modeling technique and coalition formation algorithm that forms robust and stable coalitions for task execution. This addresses some issues in many current approaches to coalition formation, namely the lack of provisions for uncertainty. Addressing this enables application of coalition formation techniques to complex domains, such as real-world robotic systems. The approach additionally provides ability to handle dynamic addition or removal of agents from the collective without constraining the environment

to limit the sources of uncertainty. The agent modeling approach enforces stability, allows for arbitrary expansion of the collective, and serves as a basis for calculation of individual coalition payoffs. It explicitly captures uncertainty in agent type and allows uncertainty in coalition value and agent cost, and no agent in the collective is required to perfectly know another agents type. The modeling approach is incorporated into a two part algorithm to generate, evaluate, and join stable coalitions for task execution. In describing the scalability, the applicability to real-world systems, and the general strength (via comparison) of the approach, this dissertation shows that coalition formation is applicable to real-world systems and is effective in the face of uncertainty.

7.1 Suggestions for Future Work

The expansion of coalition formation to robotic collectives is a difficult problem to solve, since the robots have to address many real-world constraints that agents do not. Though this dissertation solves both this problem and the problem of applying coalition formation to arbitrary collectives, there is much that can be done to expand the approach.

- Incorporate a direct capturing of agent cost uncertainty. The uncertainty in agent cost is handled by the current approach, but it is not captured. Explicitly capturing uncertainty in agent cost would expand the representation to pass uncertainty through to the coalition value itself. This would enable the generation of a confidence interval on a coalition. The agents can then make more educated decisions about potential coalitions, since they can have some confidence about a minimum payoff.
- Create a robust, generalized approach to task execution with arbitrary agent types. The allocation method presented here allows for arbitrary agent types, but the allocation is basically an academic exercise without mechanisms for executing the task. For the execution in Chapter VI, a task execution approach

which tracks progress [27] is implemented. This is an approach that applies to arbitrary agent types, but is not robust nor is it generalized.

- Apply suboptimal algorithms for coalition formation to the stochastic model. This would speed up allocation of robots to tasks, but some optimality would be lost. This would make the scalability of the approach much greater, providing solutions to task allocations in a much shorter time. The suboptimal algorithm would be well suited to large collectives.
- Expand the representation to include external resources. The only "shared" resources in the current approach are within an agent, between the Proposer and Responder. An approach designed to incorporate external resources may increase the odds of deadlock, but it would make available certain other aspects, such as movable external sensors, thereby increasing the abilities of a robot collective.
- Expand the model propagation to incorporate internal dynamics. This allows agents to incorporate a time-based degradation on the quality of an estimate of another agent's type. This expansion would, in essence, allow an agent to "forget" other agents over time. This could be useful for applications where certain collective members rarely interact and the abilities or hardware of one member can change in the time between interactions.
- Apply a different estimation technique for the model update. The straightforward Kalman filter based approach applied in this dissertation assumes linearity. Under certain variants of the approach presented here, the linearity assumption may prove invalid. An extended Kalman filter, an unscented Kalman filter, or a particle filter may prove a more effective estimation technique for those variants.
- Apply the model aggreement and propagation methodology to a trust-based agent system. Since trust does not drift, the ability of a trust network to remain robust over time is very high. However, trust-based systems are not often capable of catching a corruption immediately after it occurs. With the

model agreement approach presented in this dissertation, the agent systems would have an additional tool with which to verify their expectations about other agents. When another agent changes significantly (becomes corrupted), it informs the other agents of something different from what the other agents expect. This enables detection of a corruption as it occurs, and further improves the robustness of a trust-based agent system.

Appendix A. Deadlock and Processing Time Evaluation

This appendix provides an in-depth investigation into deadlock. It shows that deadlock is not possible under the framework provided in this dissertation by describing all of the possible sources of deadlock and how the risks are mitigated. Additionally, this appendix describes the runtime of the framework. It provides worst-case scenarios and analyzes them, showing that the scalability is at least as good as other "brute force" algorithms for coalition formation.

Deadlock is a possiblity in every multiagent system unless steps are taken directly to mitigate the risks. There are five potential sources of a deadlock in the context of the procedure that has been developed: a process waiting on a process, which is waiting on another process (process-process); a process waiting on communication, with the communication waiting on a process (process-communication); a communication waiting on communication (communication-communication); where there are multiple winning coalitions for a task (agreement); and when an agent waits on responses that never come (no response).

A.1 Deadlock Evaluation

For each of the five potential sources of deadlock, the four deadlock conditions must be met for the system to be in a deadlock. The four conditions are:

1. Mutual exclusion: a resource cannot be used by more than one process at a time.
2. Hold and wait: processes already holding resources may request new resources.
3. No preemption: No resource can be forcibly removed from a process holding it and resources can be released only by the explicit action of the process.
4. Circular wait: processes form a circular chain where each process waits for a resource that the next process in the chain holds.

A.1.1 Process-Process Deadlock. Within an agent, the only two simultaneously running processes are the proposer and the responder. In this, the only shared

data (within an agent) is in whether a coalition is formed, the ability sets for the agents, and the set of potential coalitions. The ability sets are accessed at one specific time for each procedure (when the required abilities are being generated), and no other shared resources are used during this time. Similarly, the set of potential coalitions are accessed only once for the responder and twice for the proposer, and the resource is held only long enough to complete the process. No other shared resources are accessed during this time. Finally, the formation of a coalition follows the same pattern: no other shared resources are used while the formed coalition is accessed. All of these resources defeat the hold and wait property of deadlock since each is accessed independently of the others. Even with scaling to more agents, the processes do not enter deadlock (though they may be more delayed).

A.1.2 Process-Communication Deadlock. The only time an agent waits on a communication before continuing processing is when it proposes a coalition. The proposer, however, can be preempted by the responder if it receives the same coalition proposal from another agent. The responder operates as usual, but with the data saved already the proposer, so it reduces the responder's workload. If approved, the proposed coalition is saved in the list of possible coalitions that the proposer makes use of later. The proposer ceases its wait and continues with the next coalition. The worst case of this is when every agent proposes a coalition at the same time. However, since every communication message is sent with a timestamp, the proposer whose broadcast was sent with the smallest value for the timestamp takes priority and maintains the proposer role. If timestamps are equal, the proposer role is given to the agent with a lower number designation (i.e. agent 1 assumes the proposer role if agent 2 also proposes the coalition with the same timestamp). Thus, no matter the role, the formation request is not deadlocked. With the preemption and the nature of the formation request format, there is no possibility for process-communication deadlock.

A.1.3 Communication-Communication Deadlock. There is no time where communication is waiting on communication, since communication only waits on a process or vice versa. The closest situation to when a communication is waiting on a communication is in the Process-Deadlock scenario described previously. A communication-communication deadlock is not possible given this framework.

A.1.4 Agreement Deadlock. An agreement deadlock assigns multiple coalitions to a specific task. The coalitions for a given task are disjoint: no member of one of the coalitions is in any of the other coalitions. To ensure that the tasks are only assigned to a single coalition, winning coalitions announce their formation to all the other agents. The proposer makes the announcement, and the proposer with the most profitable coalition is the proposer whose coalition is assigned to the task. This creates a social rule to mitigate the potential agreement deadlock and a priority based upon internal clock. If two agents tie for the most profitable coalition, preference is given to the agent with the lowest agent number.

A.1.5 No Response Deadlock. If no response is received when an agent submits a proposal, there are three possible causes. The first cause is if the agent that should have received the proposal does not receive it. The second cause is if the agent on the receiving end malfunctioned. The third cause is if the response is simply not received by the proposer. From the proposer's perspective, it is impossible to discern between these types of failures. Thus, to ensure ample time has been provided for a response to be received by the proposer, a three second delay is used between requests. If the response is not received within this time, the request is resubmitted with the same timestamp used on the original message. If 10 additional attempts are made without a response, the proposer gives up, removes the proposed coalition from the list of potential coalitions, and does not add the coalition to the list of approved coalitions. This prevents a deadlock since the agent has some point at which it gives up. If the agent does give up, the coalition is removed from only the

proposer agent. Other agents may propose this coalition at a later time with similar or different results, depending upon the communication situation at the time.

A.2 Communication Evaluation

The worst case communication scenario at a snapshot of time in a large collective is when a single agent dissents and responds last to a proposal involving a large coalition while other large coalitions are proposed. Recall that a single "no" response defeats a proposal. Additionally, the "no" response is sent to each agent in a proposed coalition, so any further "yes" responses need not be sent and are irrelevant. Thus, the worst case scenario is when the dissenter responds last. A specific instance of this scenario may be shown with a large number of agents arranged in a circle. Every agent proposes a coalition involving all the agents except for the agent to the proposer's right. Each proposed coalition is unique. The saturation on behalf of the sole dissenter is most significant if it is the same one and it responds last, with the coalition it proposes being declined by a single dissenter, also responding last. This is illustrated in Figures 1.1 through 1.4 with 8 agents. In this representation, n is the number of the agents in the collective and p is the number of agents in a coalition.

In Figure 1.4, 1 responds negatively to 8 and 8 responds negatively to every agent. These are broadcast to every potential coalition member, so 8 responds negatively 6 times, broadcasting each time (36 total responses). 1 responds to 8's proposed coalition once, to each agent (6 times). Note that this system is a snapshot: the number of proposed coalitions (assuming every coalition is proposed) for the entire time is every coalition but the empty set and the singletons: $2^n - 1 - n$. These $2^n - 1 - n$ coalitions are proposed once, and responded to $p - 1$ times for approval and $p - 2 + p - 1 = 2p - 3$ times for worst case disapproval, where p is the size of the coalition.

A.2.1 Communication Message Evaluation: Snapshot. In the eight agent system shown, each agent sends one proposal with 6 messages $(n - 2)$. Agents 2-7

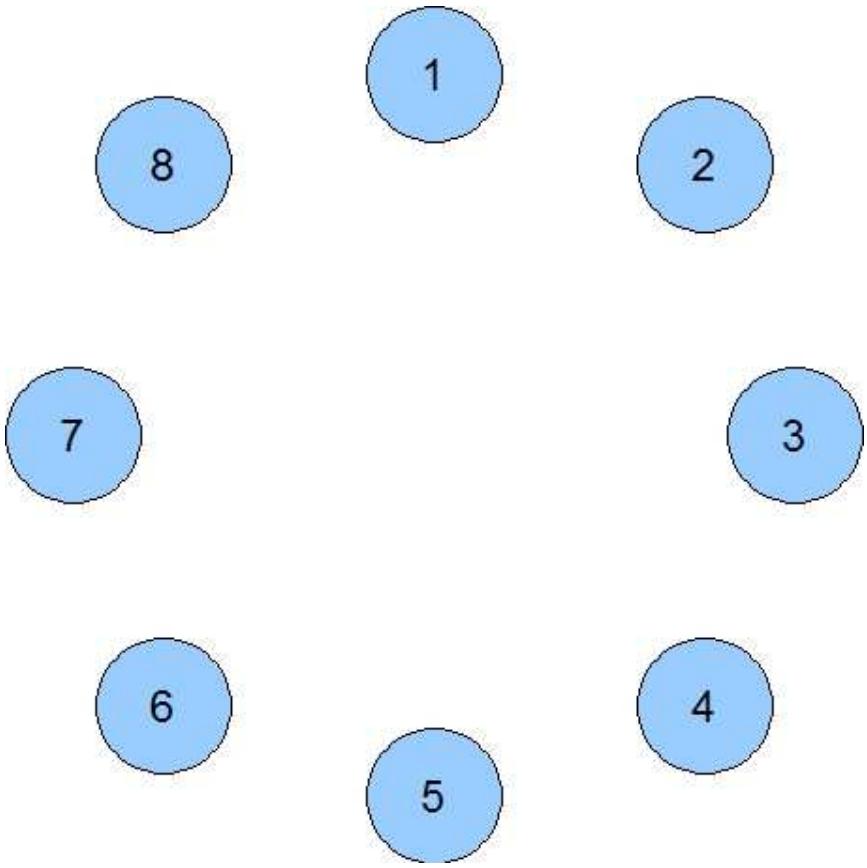

Figure 1.1: 8 agents, arranged in a circle.

($n-2$ total agents) respond positively 6 times ($n-2$). Agent 1 responds positively 5 times ($n-3$) and negatively once, sending 6 messages ($n-2$). Agent 8 responds negatively 6 times ($n-2$), sending 6 messages each time ($n-2$), making 36 ($(n-2)^2$) total messages. In addition, each agent broadcasts its quality to every other agent before the procedure begins ($n(n-1)$). The total messages sent during this worst-case scenario is $n-2+(n-2)(n-2)+n-3+n-2+(n-2)2+n2-n = 3n^2-7n-6$. Adding another agent to the mix causes this to grow to $3n^2-n-10$. This is a difference of $6n-4$ messages if a new agent is added under the worst case scenario. In the eight agent system, this is the equivalent of adding 44 communication messages. In general, the addition of another agent adds $2n-1$ more messages for the dissenter, $2n-1$ more for the new agent, and one more for each of the other remaining agents

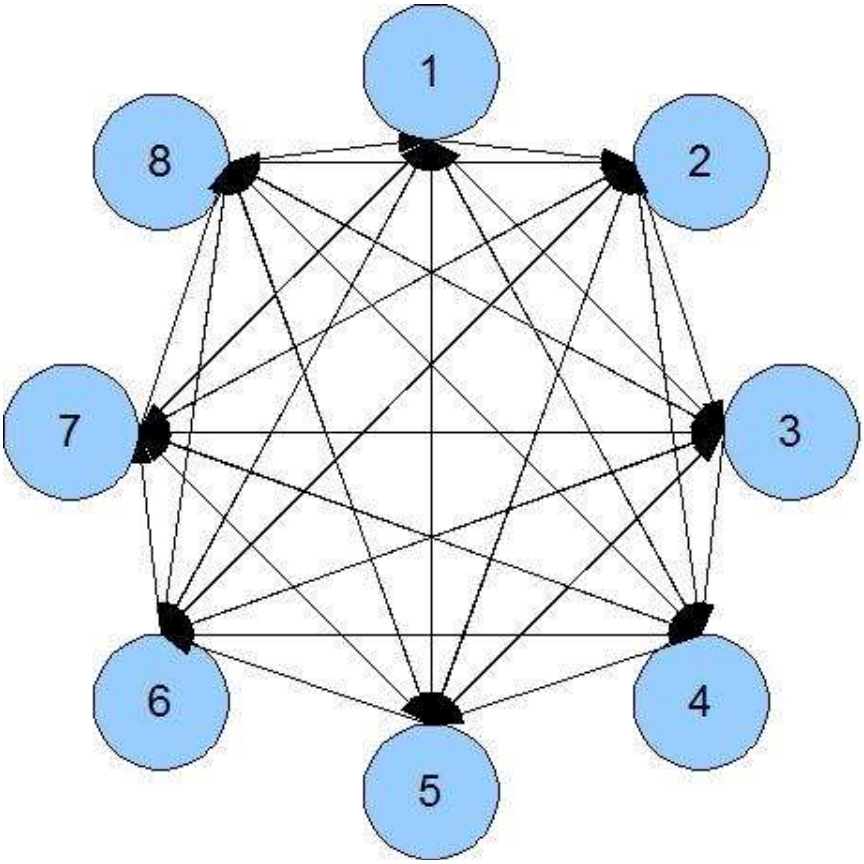

Figure 1.2: 8 agents, issuing proposals. Each proposal is sent to every agent except the agent to the proposer's right.

$(n-1)$. The agent which dissents the dissenter's proposal adds $n-1$ more on top of its one additional message. This snapshot scenario is dominated by n^2 for large n.

A.2.2 Communication Message Evaluation: Full-Term. The snapshot worst case scenario presented previously captures a very short time in the grand scheme of the coalition formation mechanism, but provides insight into the full-term behavior. The above scenario is a worst case with a single proposal per agent. However, since there are $\binom{n}{2}$ coalitions with 2 agents, $\binom{n}{3}$ coalitions with 3 agents, etc., the scale is more significant. Of these coalitions, each agent has $\binom{n-1}{1}$ coalitions of size 2 that contain itself, $\binom{n-1}{2}$ coalitions of size 3 that contain itself, etc. A reasonable estimate is to assume that each agent must propose about $\binom{n}{p}/n$ coalitions for each coalition of size p, or $\frac{2^n-1}{n}$ in total. This makes sense intuitively, since it is the total number of

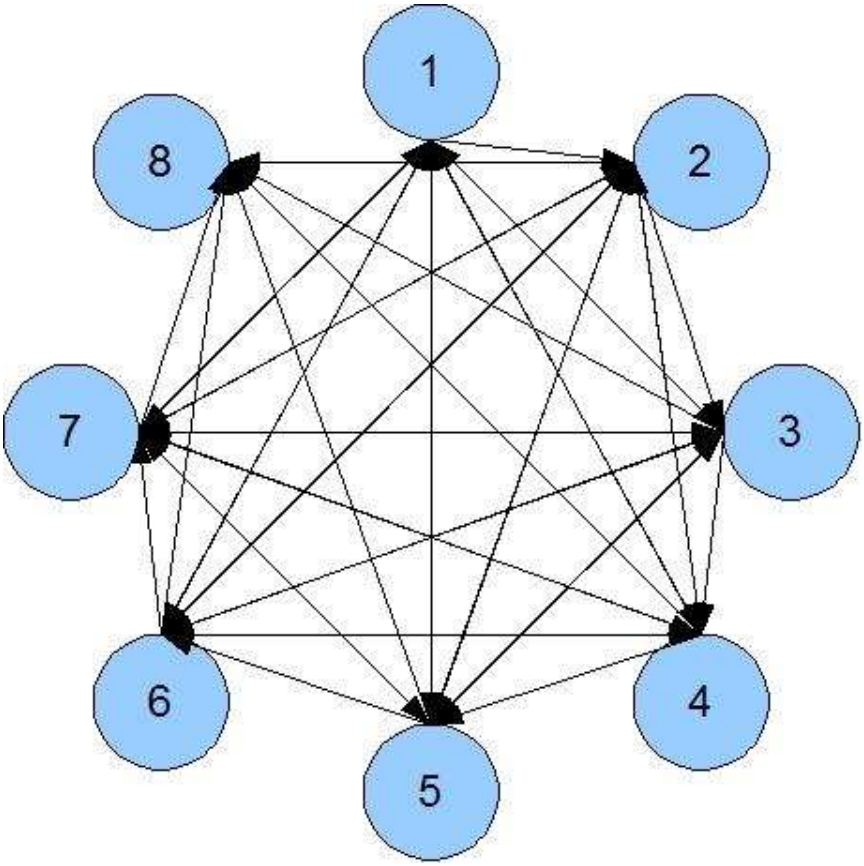

Figure 1.3: 8 agents, positive responses. Every agent except agent 8 responds positively, and agent 1 does not respond positively to agent 8.

nonempty coalitions scaled by the number of agents in the collective. Given this and the fact that each agent can be in one of $2^{n-1}-1$ possible coalitions, then each agent must respond to each coalition it does not propose, about $2^{n-1}-1-\frac{2^n-1}{n}$ proposals. In this worst case, each agent must disagree with each coalition it does not propose, and send each disagreement at the exact same time as all other non-proposer members of the coalition. This is the same context as the snapshot evaluation, but based upon the full formation time. Each agent's negative response is sent to every other member of the proposed coalition. There are the same number of coalitions of size 2 for each agent as there are of size n-1: $n-1$. The average response quantity is $\frac{(n-1)+(n-1)(n-1)}{2(n-1)} = \frac{n}{2}$. Thus, since each agent need not send a response to itself, the average response quantity is $\frac{n}{2}-1$. Therefore, each agent must send $(2^{n-1}-1-\frac{2^n-1}{n})(\frac{n}{2}-1)$ total responses. The

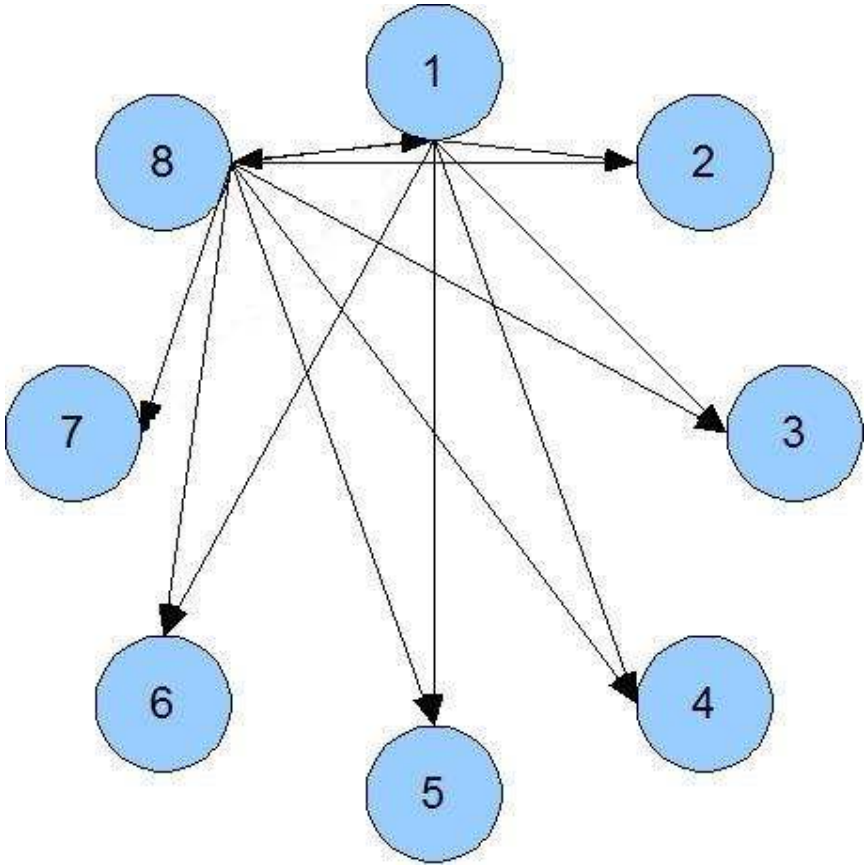

Figure 1.4: 8 agents, negative responses.

total number of messages in this worst case scenario is the sum of the responses and the proposals: $(2^{n-1} - 1 - \frac{2^n-1}{n})(\frac{n}{2} - 1) + (\frac{2^n-1}{n})(\frac{n}{2} - 1)$, which is dominated by $n * 2^n$ at large n. With communication failures, the worst case is when each agent re-sends a proposal 9 times to every agent in its proposed coalition, every agent downstream receives it and responds, but the communication fails on response. The 10th attempt is successful. Then the total message quantity is 10 times higher than previously mentioned, giving $(10)((2^{n-1} - 1 - \frac{2^n-1}{n})(\frac{n}{2} - 1) + (\frac{2^n-1}{n})(\frac{n}{2} - 1))$.

This scenario only considers the negotiation phase: the true communication quantity includes $n - 1$ messages each for announcement of qualities, a quantity of duplicate requests due to communication failure, some messages passed for final approval, and up to $n - 1$ messages passed to announce a winning coalition. This then drives the communication to $2n - 2$ more messages than previously mentioned,

or $(10)((2^{n-1}-1-\frac{2^n-1}{n})(\frac{n}{2}-1)+(\frac{2^n-1}{n})(\frac{n}{2}-1))+2n-2$. This is still dominated by $n*2^n$ at large n, so the order of the message quantity is unchanged.

A.2.3 Best Case Communication Scenario. If perfect knowledge is available, each agent can determine, without feedback from other agents, whether a coalition will be acceptable. The scope of acceptable coalitions may then be much smaller than the generated coalitions since the agents can filter their results prior to proposing. Consider a 6 agent system where two agents are poor at a task, two are moderate, and two are good. The poor agents would not agree to any proposal, so they submit none. The moderate agents would know that the poor agents would not agree, so they generate $2^{4-1}-1=7$ coalitions each instead of $2^{6-1}-1=31$ coalitions. There are a total of 11 coalitions: $\binom{4}{2}+\binom{4}{3}+\binom{4}{4}=6+4+1=11$. Each agent is likely to propose about 3 of their coalitions, all of which are acceptable. with a total of 34 messages to be sent, each agent should have about 8.5 communications. Rounding up gives 9 communication messages per agent, as opposed to a worst case of $(2^{n-1}-1-\frac{2^n-1}{n})(\frac{n}{2}-1)+(\frac{2^n-1}{n})(\frac{n}{2}-1)=(31-\frac{63}{6})(2)+(\frac{63}{6})(2)=62$. Both of these values include only the communications from the evaluation stage. The true communication quantity includes 5 messages each for announcement of qualities, a quantity of duplicate requests due to communication failure, some messages passed for final approval, and up to $n-1$ messages passed to announce a winning coalition. This analysis must be performed on a case-by-case basis, since the effects of full knowledge cannot be evaluated without specifics about the number of agents that will disagree with a proposed coalition. Under the assumption of full state knowledge, however, the number of communication messages may drop dramatically. If the agents merely have accurate models (and not full state knowledge), the assumption cannot be used and the filtering of potential proposals does not occur. If this is the case, the communication is no better nor worse than any other circumstance without full state knowledge.

A.3 Processing Evaluation

In the worst case processing scenario, each agent must calculate every possible coalition that involves itself. There are $2^{n-1}-1$ of these coalitions for each agent. The profit from each coalition is also evaluated relative to the singleton coalition, and the coalition is discarded if the coalition is less profitable. The worst case is when every coalition except the grand coalition is saved. Furthermore, each coalition involving other agents may be proposed by one of these other agents. Each agent will not have to propose every calculated coalition (since other agents will have duplicates), but let's assume, for the worst case, that the coalitions are being proposed at the exact same time by every agent involved. One proposer is maintained. Thus, each agent calculates their $2^{n-1}-1$ coalitions and proposes them all. This requires $p(p-1)$ calculations per coalition of size p: once through the coalition to generate the ability sets ($p-1$ time) and once through to generate the profit vector (p time). Assume each agent retains the proposer role for $\frac{2^n-1}{n}$ of their proposals. Then the proposer must sort each of these proposals so it can propose the most profitable coalitions, taking $\frac{2^n-1}{n}\log(\frac{2^n-1}{n})$ time. The remaining proposals are handled by the responder role, which also requires $p(p-1)$. The overall proposer time is $\sum_{p=1}^{n}\binom{n}{p}\frac{p(p-1)}{n} + \frac{2^n-1}{n}\log(\frac{2^n-1}{n}) = (n-1)(2^{n-2}) + \frac{2^n-1}{n}\log(\frac{2^n-1}{n})$, including sorting. Add to this the responder role, $\sum_{p=1}^{n}(\binom{n-1}{p-1}-\binom{n}{p}\frac{1}{n})(p(p-1))$. This gives $2^n-(n+1)$ for the responder and $(n-1)(2^{n-2})+\frac{2^n-1}{n}\log(\frac{2^n-1}{n})+2^n-(n+1)$ overall, which simplifies to $2^{n-2}(n+3)-n-1+\frac{2^n-1}{n}\log(\frac{2^n-1}{n})$. This is dominated at high n by $n2^{n-2}$, which is of the same scale as the worst case communication scenario.

A.4 Relationship Between Communication and Processing

The worst case communication scenario is dominated by $10n2^n$, whereas the worst case processing scenario is dominated by $n2^{n-2}$ at large n. This means that roughly 40 times as many communication messages are sent in the worst case than the number of processes that are executed. The growth rate is the same, as both are $\mathcal{O}(n2^n)$. This is highly inefficient in terms of time of processing relative to the

number of agents, but is similar to optimal coalition formation algorithms. Since the processing is decentralized, the workload handled by each agent is reduced versus a single centralized approach, even though duplicate work is being performed. According to Shehory and Krauss [35], finding the optimal coalition structure for a task takes $\mathcal{O}(n^n)$. However, since we're restricting the task to having only one coalition, a full structure consists of only a single coalition and the processing is reduced. Their algorithm takes $\mathcal{O}(n^{n-1})$ in computation complexity and $\mathcal{O}(n^{n-1})$ in communications complexity. The procedure developed here is better than theirs with respect to these, if only because it is solving for a single coalition instead of the entire structure. More significantly, the procedure developed here allows for uncertainty, whereas Shehory and Krauss' algorithm requires each agent to have full state knowledge.

The relative growth of the processing and communication as the number of agents increase does not cause one to swamp the other. This allows for consistent, if slowed, communication and processing growth rates and the odds of having a backlog that must be cleared in either respect is reduced. The coalition formation process takes longer with more agents, but this is expected since there are more potential coalitions.

A.5 Effects of Existing Tasks

The tasks are processed individually and sequentially. If the tasks are independent, a coalition is developed for each individually, processed one at a time. If they are not independent, they are presented as subtasks and a single coalition is developed that handles every subtask. The task presentation method and evaluation of task dependence is a decision made by the user. At the lowest level, dependent tasks are collected into a single task that represent all required aspects. This conglomerated task is then processed as any independent task would be. The sequential processing of tasks forces agents to be able to process variations in their taskability whenever they're occupied. The agents have a representation of skill qualities for when the abilities of the agent are occupied with another task. The tasks are described in a

way that allows the agents to note conflicts to concurrent possible task assignments and modify their qualities and the expectations of occupied agents appropriately. With this and the effects of computational load excepted, there are no changes to the coalition formation aspects when there are existing tasks.

Appendix B. Kalman Filter Construction

The Kalman filter variant as presented in Chapter III provides the mechanics of the belief update procedure. This appendix provides a concrete example of its application for additional reference.

Consider two agents. These agents, agents 1 and 2, are of types 3 and 1 as they were presented in Chapter III, respectively. The effects of agent 2 modeling agent 1 are investigated. Agent 1's abilities and ability values are: ALL_STOP: 0.8, MOVE_ITEM: 0.6, GOTO_XY: 0.6, FLIGHT: 0.9, OBSTACLE_AVOID: 0.9, RANGE: 0.8, GRAY_VISION: 0.9, and PRECISION_NAV: 1.0. For a task to move an item, the abilities used are ALL_STOP, MOVE_ITEM, GOTO_XY, FLIGHT, OBSTACLE_AVOID, and RANGE. Assume, for this example, that the agents send the mean value of the abilities they plan to use for the task. Agent 1 therefore sends a value of 0.767 for its quality.

Agent 2 has the abilities ALL_STOP, GOTO_XY, LIFT, MOVE_ITEM, OBSTACLE_AVOID, RANGE, and RGB_VISION. It assumes Agent 1 has the same set of abilities. It then expects the use of ALL_STOP, GOTO_XY, LIFT, MOVE_ITEM, OBSTACLE_AVOID, and RANGE to execute the task. Assume the ability qualities were initialized at 0.5 for all abilities, and all abilities are mutually independent, i.e., all nonzero values in the covariance matrix are on the diagonal. Also assume the initial values for the variance of the ability values are 0.5. The filter values x, P, and H are then defined as:

$$x_0 = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}$$

$$\mathbf{P}_0 = \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix}$$

$$H = \begin{bmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 \end{bmatrix}$$

Let $x_{0^+} = x_0$. Then with the dynamics of the system at steady state, $x_{1^-} = x_{0^+}$. Likewise, $\mathbf{P}_{1^-} = \mathbf{P}_{0^+} = \mathbf{P}_0$. The residual

$$y_1 = 0.767 - \begin{bmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix} = 0.267$$

Define v_k as zero mean white Gaussian nose with variance 0.1. Then the residual covariance is

$$\mathbf{S}_1 = \begin{bmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 \end{bmatrix} \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix} \begin{bmatrix} \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ 0 \end{bmatrix} + 0.1 = 0.2833$$

with Kalman gain of

$$K_1 = \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix} \begin{bmatrix} \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ 0 \end{bmatrix} \left(\frac{1}{.2833} \right) = \begin{bmatrix} 0.2941 \\ 0.2941 \\ 0.2941 \\ 0.2941 \\ 0.2941 \\ 0.2941 \\ 0 \end{bmatrix}$$

Giving

$$x_{1^+} = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix} + \begin{bmatrix} 0.2941 \\ 0.2941 \\ 0.2941 \\ 0.2941 \\ 0.2941 \\ 0.2941 \\ 0 \end{bmatrix} (0.267) = \begin{bmatrix} 0.5784 \\ 0.5784 \\ 0.5784 \\ 0.5784 \\ 0.5784 \\ 0.5784 \\ 0.5000 \end{bmatrix}$$

and

$$\mathbf{P}_{1^+} = \left(I_7 - \begin{bmatrix} 0.2941 \\ 0.2941 \\ 0.2941 \\ 0.2941 \\ 0.2941 \\ 0.2941 \\ 0 \end{bmatrix} \begin{bmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 \end{bmatrix} \right) \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix}$$

$$= \begin{bmatrix} 0.4755 & -0.0245 & -0.0245 & -0.0245 & -0.0245 & -0.0245 & 0 \\ -0.0245 & 0.4755 & -0.0245 & -0.0245 & -0.0245 & -0.0245 & 0 \\ -0.0245 & -0.0245 & 0.4755 & -0.0245 & -0.0245 & -0.0245 & 0 \\ -0.0245 & -0.0245 & -0.0245 & 0.4755 & -0.0245 & -0.0245 & 0 \\ -0.0245 & -0.0245 & -0.0245 & -0.0245 & 0.4755 & -0.0245 & 0 \\ -0.0245 & -0.0245 & -0.0245 & -0.0245 & -0.0245 & 0.4755 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5000 \end{bmatrix}$$

This process is repeated when the next task is evaluated. The trends of the state vector and the standard deviations are shown in Figures 2.1 and 2.2, respectively.

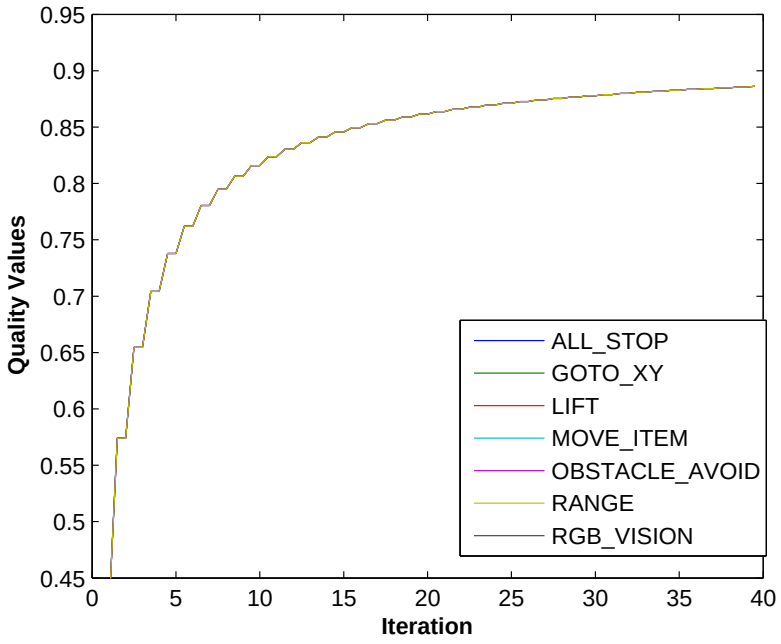

Figure 2.1: The convergence of the state vector over reevaluations of the same task.

The values for R and the initial covariances affect convergence rates. Figures 2.3 and 2.4 show the effects of setting R to 0.5. Increasing R makes an agent trust its currently existing model more, and trust the measurements less. This increases the time it takes to converge on a value. Setting the initial covariance to 0.8 has a

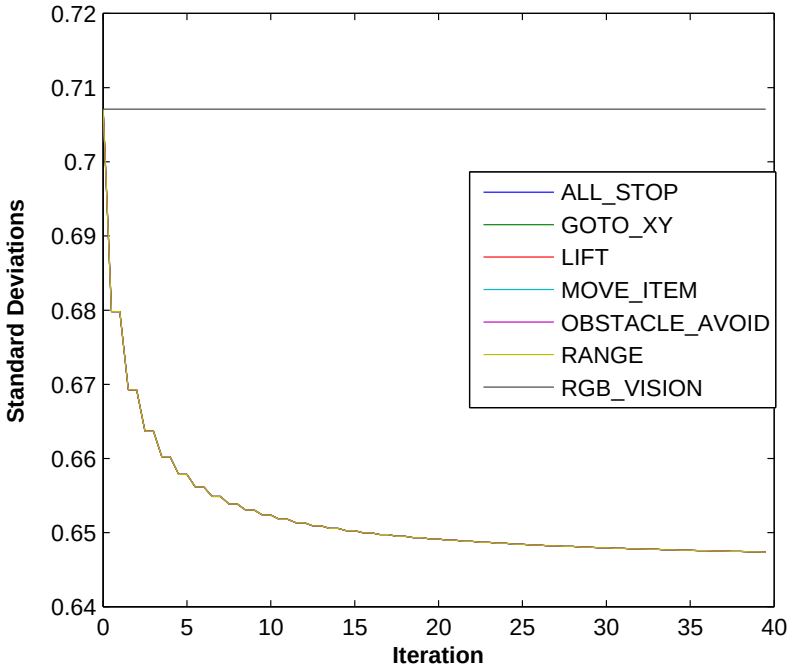

Figure 2.2: The standard deviations over reevaluations of the same task.

minor effect on the state, but a significant effect on the final covariance, changing the steady-state value. This is shown in Figure 2.5.

These results do not consider the effects of different task types. However, those task type variations are investigated in Chapter VI.

B.1 Robot Ability Qualities

The residuals of the measurements are shown in Chapter VI. This provides insight into the performance of the robot models over time, but does not actually provide the internal structure of the models. Those are presented here for clarity. Figures 2.6 through 2.8 show the models of Gandalf, Frodo, and the NXT before Gollum is added to the collective. Their models of each other are tracked, and the quality values of each ability are shown in the figures.

When Gollum is added to the collective, the other robots have well-established models of each other, so the changes in their models indicated by Figures 2.9, 2.10, and

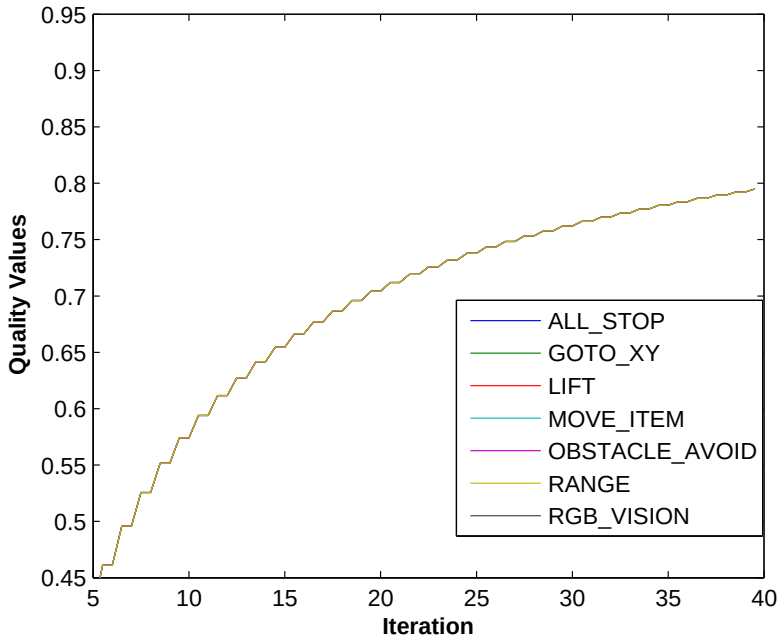

Figure 2.3: The convergence of the state vector over reevaluations of the same task. v_k has a covariance of 0.5.

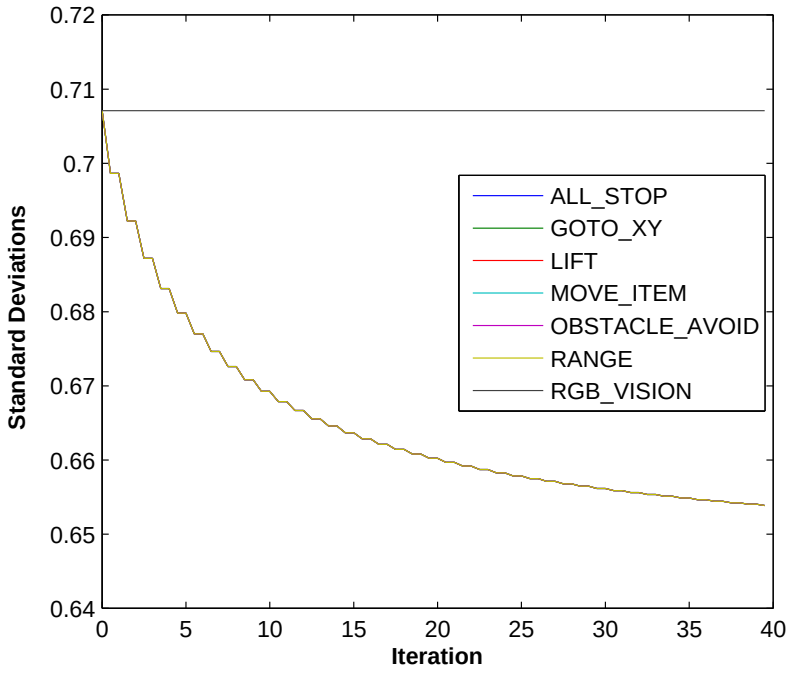

Figure 2.4: The standard deviations over reevaluations of the same task. v_k has a covariance of 0.5.

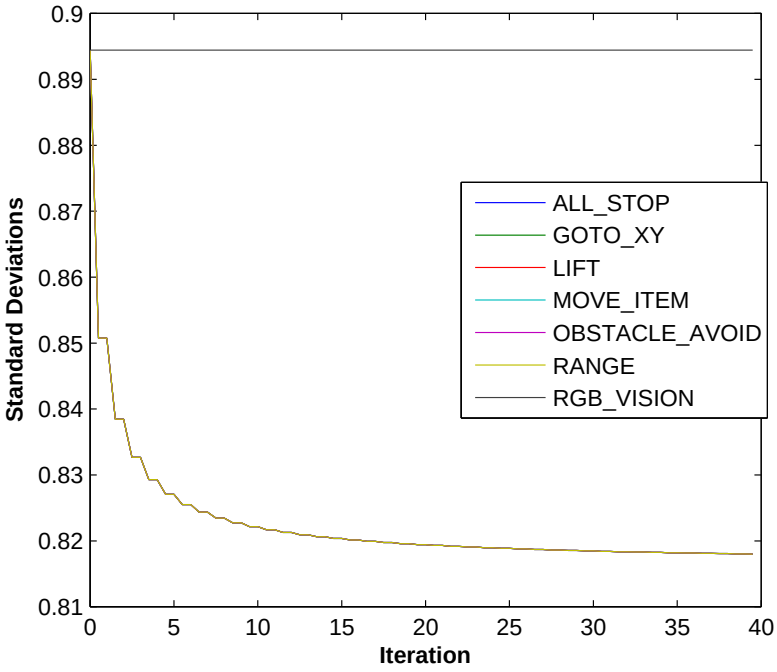

Figure 2.5: The standard deviations over reevaluations of the same task. Initial covariances are set to 0.8.

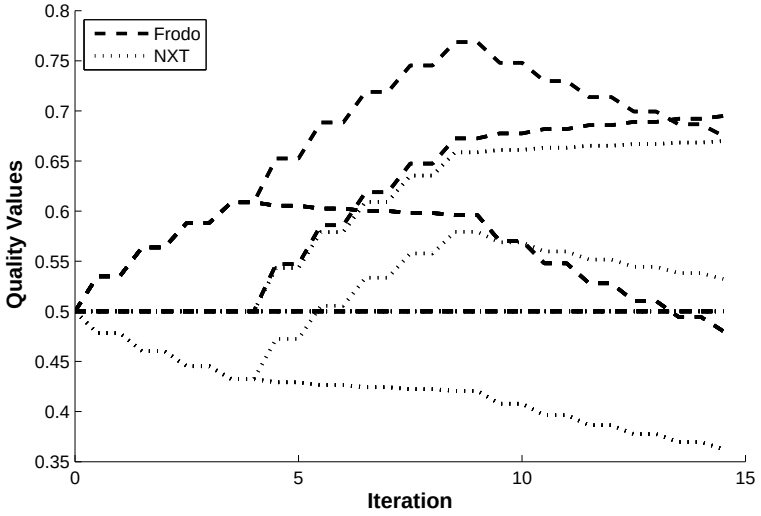

Figure 2.6: Gandalf's model changes over the 15 negotiations prior to the addition of Gollum. The branching effect occurs when abilities that were part of a previous set (used or unused) are applied again.

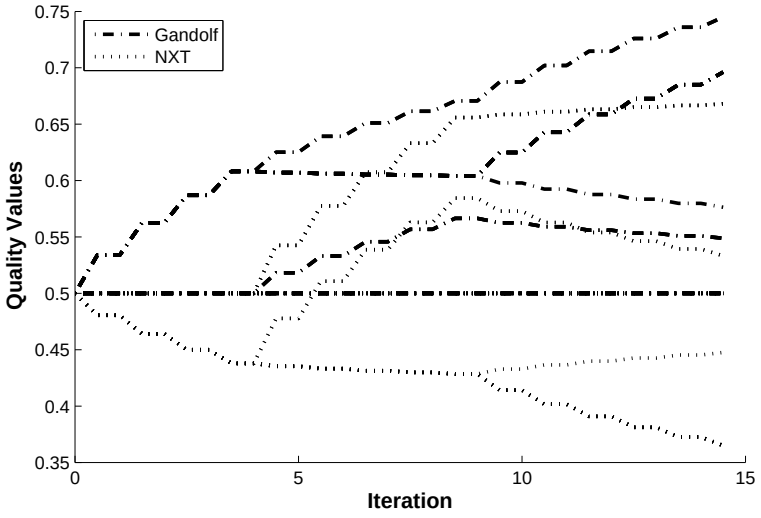

Figure 2.7: Frodo's model changes over the 15 negotiations prior to the addition of Gollum.

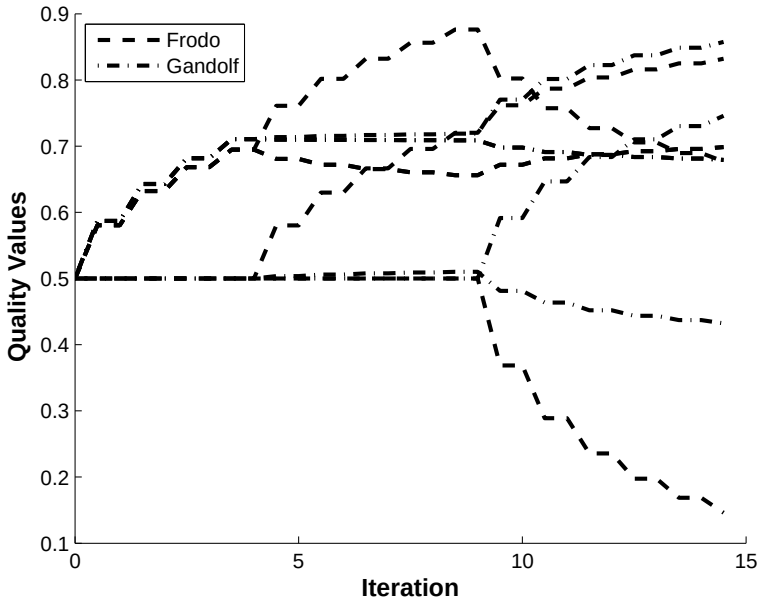

Figure 2.8: The NXT robot's model changes over the 15 negotiations prior to the addition of Gollum.

2.11 change little by comparison, with the exception of the models for Gollum. Figures 2.9, 2.10, and 2.11 also serve to indicate another strength of this representation: despite the different ability sets that the robots are working from and the different

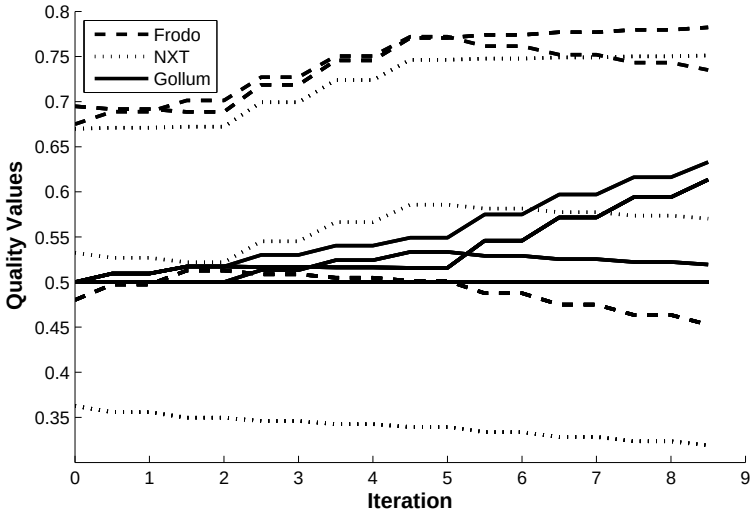

Figure 2.9: Gandalf's model changes over the 9 negotiations after the addition of Gollum.

assumptions on ability usage on a given task, Gandalf, Frodo, and the NXT robot all have models that respond in a similar manner to the quality values provided by Gollum. The NXT robot is working from a small ability set, containing only 5 types of abilities. Frodo, Gandalf, and Gollum are working off ability sets between 22 and 30 elements in size, making their representation much more sophisticated. This expanded sophistication provides only moderate gain in terms of model accuracy, but allows each robot to maintain a representation over which it can reason independently of the other robots.

Gollum's model changes over nine negotiations after being added to the collective are shown in Figure 2.12. The model results at the end of the nine negotiations reflect a similar end point as the models of Frodo, Gandalf, and the NXT robot while having less time to fully indoctrinate.

The plots in this section indicate the actual ability qualities, according to the internal representation of each robot. This is dependent upon the parameters as presented in Chapter VI, the physical hardware of the robots, the abilities each robot possesses, and the abilities each robot expects to be used for a task. These plots are presented to provide insight into the internal mechanism of the model update.

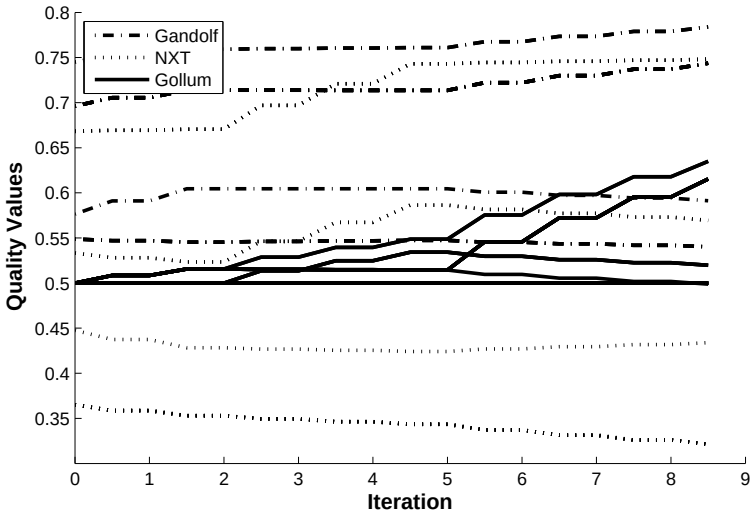

Figure 2.10: Frodo's model changes over the 9 negotiations after the addition of Gollum.

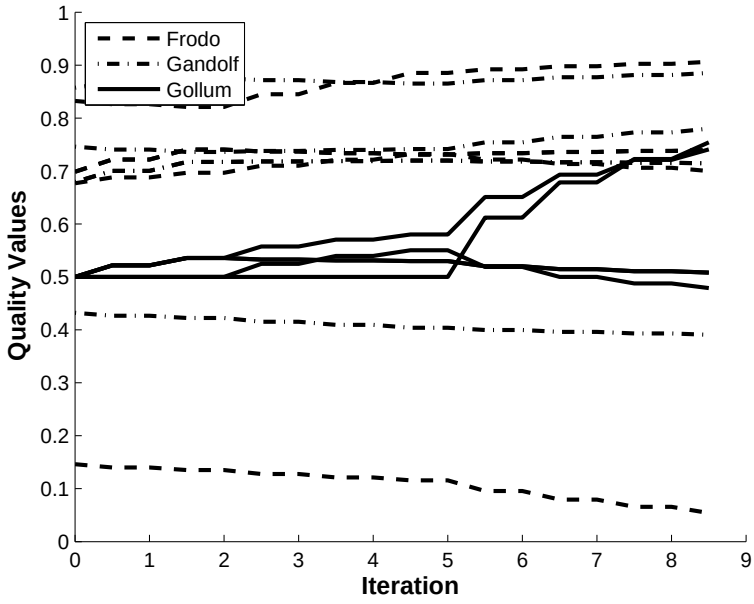

Figure 2.11: The NXT robot's model changes over the 9 negotiations after the addition of Gollum.

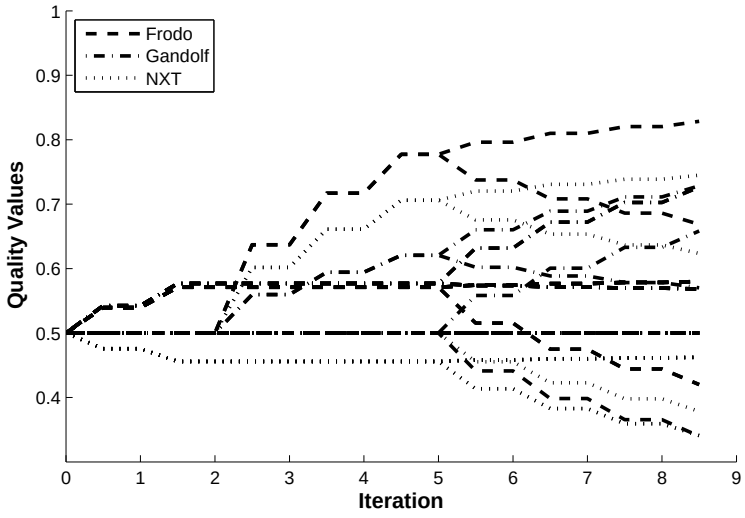

Figure 2.12: Gollum's model changes over 9 negotiations after added to the collective.

Bibliography

1. Abdallah, Sherief and Victor Lesser. "Organization-Based Cooperative Coalition Formation". *IAT '04: Intelligent Agent Technology, IEEE/WIC/ACM International Conference*, 162–168. IEEE Computer Society, Washington, DC, USA, 2004. ISBN 0-7695-2101-0.

2. Allen, Martin and Shlomo Zilberstein. "Agent Influence as a Predictor of Difficulty for Decentralized Problem-Solving". *Twenty-Second AAAI Conference on Artificial Intelligence, AAAI 2007*, 688–693. Vancouver, BC, Canada, July 2007.

3. Arkin, R. "Motor Schema Based Navigation for a Mobile Robot: An Approach to Programming by Behavior". *The 1987 IEEE International Conference on Robotics and Automation, ICRA-87*, volume 4, 264–271. March 1987.

4. Aumann, Robert J. *Collected Papers*, volume 2, chapter 39: The Core of a Cooperative Game without Side Payments, 13–26. MIT Press, 2000.

5. Aumann, Robert J. *Collected Papers*, volume 2, chapter 40: A Survey of Cooperative Games without Side Payments, 31–55. MIT Press, 2000.

6. Bernstein, Daniel S., Robert Givan, Neil Immerman, and Shlomo Zilberstein. "The Complexity of Decentralized Control of Markov Decision Processes". *Mathematics of Operations Research*, 27(4):819–840, 2002. URL `http://dblp.uni-trier.de/db/journals/mor/mor27.html#BernsteinGIZ02`.

7. Blankenburg, Bastian, Matthias Klusch, and Onn Shehory. "Fuzzy Kernel-Stable Coalitions Between Rational Agents". *Second International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS '03*, 9–16. ACM, New York, NY, USA, 2003. ISBN 1-58113-683-8.

8. Bowring, Emma, Jonathan P Pearce, Christopher Portway, Manish Jain, and Milind Tambe. "On k-Optimal Distributed Constraint Optimization Algorithms: New Bounds and Algorithms". *7th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS '08*, 607–614. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2008. ISBN 978-0-9817381-1-6.

9. Brooks, Rodney A. *Cambrian Intelligence*. MIT Press, 1999.

10. Chaimowicz, Luiz, Mario F. M. Campos, and Vijay Kumar. "Dynamic Role Assignment for Cooperative Robots". *IEEE International Conference on Robotics and Automation, ICRA '02*, 293–298. 2002.

11. Chalkiadakis, Georgios and Craig Boutilier. "Bayesian Reinforcement Learning for Coalition Formation under Uncertainty". *Third International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS '04*, 1090–1097. IEEE Computer Society, Washington, DC, USA, 2004. ISBN 1-58113-864-4.

12. Chalkiadakis, Georgios and Craig Boutilier. "Coalitional Bargaining with Agent Type Uncertainty". *2007 International Joint Conferences on Artificial Intelligence, IJCAI-07*, 1227–1232. Hyderabad, India, 2007.

13. Chalkiadakis, Georgios and Craig Boutilier. "Sequential Decision Making in Repeated Coalition Formation under Uncertainty". *7th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS '08*, 347–354. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2008. ISBN 978-0-9817381-0-9.

14. Chalkiadakis, Georgios, Evangelos Markakis, and Craig Boutilier. "Coalition Formation under Uncertainty: Bargaining Equilibria and the Bayesian Core Stability Concept". *6th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS '07*, 1–8. ACM, New York, NY, USA, 2007. ISBN 978-81-904262-7-5.

15. Contreras, Javier and Felix F. Wu. "A Kernel-Oriented Algorithm for Transmission Expansion Planning". *IEEE Transactions on Power Systems*, 15(4):1434–1440, November 2000.

16. Davis, Morton and Michael Maschler. "The Kernel of a Cooperative Game". *Naval Research Logistics Quarterly*, 12(3):223–259, 1965.

17. Dias, M Bernardine and Anthony (Tony) Stentz. "A Free Market Architecture for Distributed Control of a Multirobot System". *6th International Conference on Intelligent Autonomous Systems (IAS-6)*, 115–122. July 2000.

18. Edgeworth, Francis Ysidro. *Mathematical Psychics: an Essay on the Application of Mathematics to the Moral Sciences*. Kegan Paul, 1881.

19. Fanelli, L., A. Farinelli, L. Iocchi, D. Nardi, and G. P. Settembre. "Ontology-Based Coalition Formation in Heterogeneous MRS". *PCAR '06: Proceedings of the 2006 International Symposium on Practical Cognitive Agents and Robots*, 105–116. ACM, New York, NY, USA, 2006. ISBN 1-74052-130-7.

20. Gerkey, B.P. and M.J. Mataric. "Sold!: Auction Methods for Multirobot Coordination". *IEEE Transactions on Robotics and Automation*, 18(5):758–768, Oct 2002. ISSN 1042-296X.

21. Gerkey, Brian P. and Maja J. Mataric. "Multi-Robot Task Allocation: Analyzing the Complexity and Optimality of Key Architectures". *IEEE International Conference on Robotics and Automation, ICRA '03*, 3:3862–3868, Sept. 2003. ISSN 1050-4729.

22. Gerkey, Brian P., Richard T. Vaughan, and Andrew Howard. "The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems". *Proceedings of the 11th International Conference on Advanced Robotics*, 317–323. 2003.

23. Gillies, Donald B. *Contributions to the Theory of Games*, volume IV, chapter Solutions to General Non-Zero-Sum Games, 4785. Princeton University Press, 1959.

24. Gmytrasiewicz, P. and P. Doshi. "A Framework for Sequential Planning in Multi-Agent Settings". *Journal of Artificial Intelligence Research*, 24:24–49, 2004. URL `citeseer.ist.psu.edu/gmytrasiewicz04framework.html`.

25. Gomez, Juan Camilo. *Generalizing the Extended Core to Games with Nontransferable Utility*. Technical report, University of Minnesota, Department of Economics, November 2002. URL `http://www.econ.umn.edu/~jcgomez/ntu3.pdf`.

26. Hooper, Daylond and Gilbert Peterson. "HAMR: A Hybrid Multi-Robot Control Architecture". *22nd Florida Artificial Intelligence Research Society Conference, FLAIRS-22*. AAAI, Sanibel Island, Florida, May 19-21 2009. Submitted for publication.

27. Hooper, Daylond J. "Tightly Coupled Cooperation among Independent Agents". *Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008*, 1853–1854. July 2008.

28. Kaelbling, Leslie Pack, Michael L. Littman, and Anthony R. Cassandra. "Planning and Acting in Partially Observable Stochastic Domains". *Artificial Intelligence*, 101(CS-96-08):99–134, 1995. URL `citeseer.ist.psu.edu/kaelbling95planning.html`.

29. Kargin, Vladislav. *Uncertainty of the Shapley Value*. Game Theory and Information 0309003, EconWPA, September 2003. URL `http://ideas.repec.org/p/wpa/wuwpga/0309003.html`.

30. Luce, Robert Duncan and Howard Raiffa. *Games and Decisions: Introduction and Critical Survey*. Dover, 1989. ISBN 0-486-65943-7. Reprint. Originally published:Wiley, 1957.

31. McAfee, Preston R. and John Mcmillan. "Auctions and Bidding". *Journal of Economic Literature*, 25(2):699–738, 1987. ISSN 00220515.

32. Osborne, Martin J and Ariel Rubinstein. *A Course in Game Theory*. The MIT Press, Cambridge, Massachusetts, 1994. ISBN 978-0262650403.

33. Parker, Lynne E. "Adaptive Heterogeneous Multi-Robot Teams". *Neurocomputing*, 28:75–92, 1999.

34. Sandholm, Tuomas, Kate Larson, Martin Andersson, Onn Shehory, and Fernando Tohmé. "Coalition Structure Generation with Worst Case Guarantees". *Artificial Intelligence*, 111:209238, 1999.

35. Shehory, Onn and Sarit Kraus. "Methods for Task Allocation via Agent Coalition Formation". *Artificial Intelligence*, 101(1-2):165–200, May 1998.

36. Shehory, Onn and Sarit Kraus. "Feasible Formation of Coalitions Among Autonomous Agents in Non-Super-Additive Environments". *Computational Intelligence*, 15(3), 1999.

37. Smith, Reid G. "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver". *IEEE Transactions on Computers*, C-29(12):1104–1113, 1981.

38. Soh, Leen-Kiat and Costas Tsatsoulis. "Utility-Based Multiagent Coalition Formation with Incomplete Information and Time Constraints". *IEEE International Conference on Systems, Man and Cybernetics*, 2:1481–1486, Oct. 2003. ISSN 1062-922X.

39. Tang, Fang and Lynne E. Parker. "Distributed Multi-Robot Coalitions Through ASyMTRe-D". *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2005*, 2606–2613. Aug. 2005.

40. Tohme, Fernando and Tuomas Sandholm. "Coalition Formation Processes with Belief Revision Among Bounded Rational Self-Interested Agents". *Journal of Logic and Computation*, 43–51. 1999.

41. Vig, Lovekesh and Julie A. Adams. "Issues in Multi-Robot Coalition Formation". *Proceedings of Multi-Robot Systems. From Swarms to Intelligent Automata*, volume 3, 15–26. 2005.

42. Vig, Lovekesh and Julie A. Adams. "Multi-Robot Coalition Formation". *IEEE Transactions on Robotics*, 22(4):637–649, Aug. 2006. ISSN 1552-3098.

43. Werger, Barry Brian and Maja J. Mataric. "Broadcast of Local Eligibility for Multi-Target Observation". *5th International Symposium on Distributed Autonomous Robotic Systems*, 347–356. Springer-Verlag, 2000.

44. Winter, Eyal. "Chapter 53: The Shapley Value". Robert Aumann and Sergiu Hart (editors), *Handbook of Game Theory with Economic Applications*, volume Volume 3, 2025–2054. Elsevier, 2002. URL `http://www.sciencedirect.com/science/article/B7P5P-4FD79WM-J/2/b206b6aac6c58b5f23dad47886416298`.

45. Woolley, Brian G. and Gilbert L. Peterson. "Unified Behavior Framework for Reactive Robot Control". *Journal of Intelligent Robotics Systems*, 55(2-3):155–176, 2009. ISSN 0921-0296.

46. Zlot, Robert, Anthony (tony) Stentz, M. Bernardine Dias, and Scott Thayer. "Multi-Robot Exploration Controlled by a Market Economy". *IEEE International Conference on Robotics and Automation*, 3016–3023. 2002.

Vita

Daylond J. Hooper graduated from Fairborn High School in Fairborn, Ohio. He enlisted in the Army in July 1998. His first assignment was at Fort Bragg, North Carolina, where he was assigned to A Company, 2nd Battalion, 505th Parachute Infantry Regiment of the 82nd Airborne Division. He deployed to Albania and Kosovo in April 1999 for six months. He received an honorable discharge in July 2000. In August 2002, he enrolled in undegraduate studies at Wright State University in Dayton, Ohio where he graduated Magna Cum Laude with a Bachelor of Science degree in Biomedical Engineering in June 2006.

He received the DAGSI fellowship for Ph.D. study and enrolled in the Graduate School of Engineering and Management, Air Force Institute of Technology in June 2006. He graduated from the Master's degree program in Computer Science in December 2007. Upon completion of the Ph.D. degree, he will begin a career pursuing research in artificial intelligence.

Permanent address: 2950 Hobson Way
Air Force Institute of Technology
Wright-Patterson AFB, OH 45433

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
12–03–2010	Dissertation	Dec 2007 — Mar 2010

4. TITLE AND SUBTITLE

Coalition Formation under Uncertainty

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Daylond James Hooper

5d. PROJECT NUMBER

JON 10-194

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/DEE/ENG/10-05

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Jacob Campbell, Civ
2241 Avionics Circle
Wright-Patterson Air Force Base, OH 45433
(937) 255-6127 x4154; jacob.campbell@wpafb.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Many multiagent systems require allocation of agents to tasks in order to ensure successful task execution. Most systems that perform this allocation assume that the quantity of agents needed for a task is known beforehand. Coalition formation approaches relax this assumption, allowing multiple agents to be dynamically assigned. Unfortunately, many current approaches to coalition formation lack provisions for uncertainty. This prevents application of coalition formation techniques to complex domains, such as real-world robotic systems and agent domains where full state knowledge is not available. Those that do handle uncertainty have no ability to handle dynamic addition or removal of agents from the collective and they constrain the environment to limit the sources of uncertainty. A modeling approach and algorithm for coalition formation is presented that decreases the collective's dependence on knowing agent types. The agent modeling approach enforces stability, allows for arbitrary expansion of the collective, and serves as a basis for calculation of individual coalition payoffs. It explicitly captures uncertainty in agent type and allows uncertainty in coalition value and agent cost, and no agent in the collective is required to perfectly know another agents type. The modeling approach is incorporated into a two part algorithm to generate, evaluate, and join stable coalitions for task execution. A comparison with a prior approach designed to handle uncertainty in agent type shows that the protocol not only provides greater flexibility, but also handles uncertainty on a greater scale. Additional results show the application of the approach to real-world robotics and demonstrate the algorithm's scalability. This provides a framework well suited to decentralized task allocation in general collectives.

15. SUBJECT TERMS

Artificial intelligence; multiagent systems; multi-robot systems; coalition formation; task allocation; Kalman filtering

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Gilbert Peterson, Ph.D. (ENG)
U	U	U	UU	131	19b. TELEPHONE NUMBER *(include area code)* (937) 255–3636, x4314; gilbert.peterson@afit.edu